1

Research Note 84-56

PROGRAMMER'S MANUAL TO ACCOMPANY THE YUGOSLAV DILEMMA (A COMPUTER SIMULATION)

Kenneth W. Unger and Robert W. Swezey
Science Applications, Inc.

Paul van Rijn, Contracting Officer's Representative

Submitted by

T. Owen Jacobs, Chief
LEADERSHIP AND MANAGEMENT TECHNICAL AREA

and

Joyce L. Shields, Director
MANPOWER AND PERSONNEL RESEARCH LABORATORY

U. S. Army
Research Institute for the Behavioral and Social Sciences

February 1984

Approved for public release; distribution unlimited.

This report, as submitted by the contractor, has been cleared for release to Defense Technical Information Center (DTIC) to comply with regulatory requirements. It has been given no primary distribution other than to DTIC and will be available only through DTIC or other reference services such as the National Technical Information Service (NTIS). The views, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other official documentation.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER Research Note 84-56	2. GOVT ACCESSION NO. AD-A141	3. RECIPIENT'S CATALOG NUMBER 776

4. TITLE (*and Subtitle*)

PROGRAMMER'S MANUAL TO ACCOMPANY THE YUGOSLAV DILEMMA (A COMPUTER SIMULATION)

5. TYPE OF REPORT & PERIOD COVERED

November 1983

6. PERFORMING ORG. REPORT NUMBER

SAI-83-08-178

7. AUTHOR(s)

Kenneth W. Unger and Robert W. Swezey

8. CONTRACT OR GRANT NUMBER(s)

MDA 903-79-C-0699

9. PERFORMING ORGANIZATION NAME AND ADDRESS

Behavioral Sciences Research Center
Science Applications, Inc.
1710 Goodridge Drive, McLean, VA 22102

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

2Q162722A791

11. CONTROLLING OFFICE NAME AND ADDRESS

U.S. Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

12. REPORT DATE

February 1984

13. NUMBER OF PAGES

123

14. MONITORING AGENCY NAME & ADDRESS(*if different from Controlling Office*)

15. SECURITY CLASS. (*of this report*)

UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (*of this Report*)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (*of the abstract entered in Block 20, if different from Report*)

18. SUPPLEMENTARY NOTES

This research was technically monitored by Dr. Paul van Rijn of the Leadership & Management Technical Area of ARI.

19. KEY WORDS (*Continue on reverse side if necessary and identify by block number*)

Programmer's Manual	Decision Making
Software	Cognitive Complexity
Documentation	Computer Simulation

20. ABSTRACT (*Continue on reverse side if necessary and identify by block number*)

This Programmer's Manual accompanies a management assessment and training simulation system that assesses participants' decision-making style. The manual 1) describes all hardware and software required to run the simulation, 2) identifies key variables and describes the functions of program components, 3) describes how to manipulate variables in order to modify the simulation and 4) describes how participants' responses are scored.

FOREWORD

This document is one in a series which reports on research conducted by the Behavioral Sciences Research Center at Science Applications, Inc., under Contract No. MDA 903-79-C-0699 with the U.S. Army Research Institute for the Behavioral and Social Sciences. The work on this contract has involved designing and developing a management assessment training and simulation system (MATSS), which includes a computer simulation called the "Yugoslav Dilemma," used to assess the decision-making strategy employed by executive level managers. Decision making has been found to be one of the most prevalent factors in organizational management. The major documents produced by this project include:

Swezey, R. W., Streufert, S., Criswell, E. L., Unger, K. W., and van Rijn, P. Development of a computer simulation for assessing decision-making style using cognitive complexity theory. (SAI Report No. SAI-84-04-178) McLean, VA: Science Applications, Inc., 1984.

This report is the project final report. It describes the history of the project, theoretical (cognitive complexity theory) rationale for the simulation and its assessment measures, and a complete description of the simulation. Interested readers should refer to this report for an overview and description of the project.

Baudhuin, E. S., Swezey, R. W., Foster, G. D., and Streufert, S. An empirically derived taxonomy of organizational systems. (SAI Report No. SAI-80-091-178) McLean, VA: Science Applications, Inc., 1980.

This document describes the factor analytic procedures used to cluster and rank-order over 350 variables involved in systems theory and organizational management. The procedure yielded six factors. Factor one was multidimensional information processing including decision making. This factor lead to the decision-making emphasis of the simulation.

Swezey, R. W., Davis, E. G., Baudhuin, E. S., Streufert, S., and Evans, R. A. Organizational and systems theories: An integrated review. (SAI Report No. SAI-80-113-178) McLean, VA: Science Applications, Inc., 1980.

This 300-page literature review provides an integrated discussion relating the diverse fields of organizational and systems theory. Its contents are organized according to the taxonomy developed in Baudhuin, Swezey, Foster, and Streufert (1980).

Unger, K. W. and Swezey, R. W. *Programmer's manual to accompany the Yugoslav dilemma (a computer simulation)*. (SAI Report No. SAI-83-08-178) McLean, VA: Science Applications, Inc., 1983.

This manual describes the eight programs which run the Yugoslav Dilemma. Each program is listed and annotated. Various possible program manipulations are described.

Criswell, E. L., Unger, K. W., Swezey, R. W., and Streufert, S. *Researcher's manual to accompany the Yugoslav dilemma (a computer simulation)*. (SAI Report No. SAI-84-02-178) McLean, VA: Science Applications, Inc., 1984.

The manual 1) explains the researcher's responsibilities in running participants through the simulation, 2) describes all materials necessary to operate the simulation, 3) provides step-by-step operating procedures, and 4) presents instruction for interpreting participant profiles.

Criswell, E. L., Unger, K. W., and Swezey, R. W. *Participant's manual to accompany the Yugoslav dilemma (a computer simulation)*. (SAI Report No. SAI-84-03-178) McLean, VA: Science Applications, Inc., 1984.

This manual presents 1) instructions on how to interact with the computer during the simulation, and 2) fictional background information to set the stage for the Yugoslav Dilemma.

PROGRAMMER'S MANUAL TO ACCOMPANY THE YUGOSLAV DILEMMA (A COMPUTER SIMULATION)

EXECUTIVE SUMMARY

Requirement:

There is a widely recognized need to provide top level Army managers with better information and with tools to better utilize the information they have. This need exists, not only within battle situations, but also within strategic and managerial situations. Top level decision making is typically characterized by lack of complete information, multiple and conflicting objectives, high levels of uncertainty, turbulent environments, and decision outcomes that tend to be both costly and long range in their implications.

This report describes software which is utilized by a man-machine managerial assessment and training vehicle that simulates complex information processing and decision-making requirements within a senior level military management context. This vehicle is termed the Management Assessment and Training Simulation System. The Yugoslav Dilemma is a problem scenario in the system which assesses participant's decision-making strategy. This document is the programmer's manual which accompanies the Yugoslav Dilemma.

Procedure:

Software for the Management Assessment and Training Simulation System is presented. This document provides: (1) a documented listing of simulation programs, (2) instructions for manipulating key system variables, (3) a description of system hardware, and (4) a detailed example of how participants' responses are calculated.

Findings:

Software and supporting documentation presented in this report allows the Management Assessment and Training Simulation System to function as required.

Utilization of Findings:

The Programmer's Manual is a necessary tool for understanding the simulation and manipulating key variables. The intended audience of this manual is computer programmers.

ACKNOWLEDGEMENTS

The software reported in this document was developed by Clifford T. Schafer, Wise Owl Workshop, 1168 Avenida De Las Palmas, Livermore, California. The authors acknowledge Dr. Eleanor Criswell of Science Applications, Inc. for her invaluable assistance in preparing the final version of this report.

TABLE OF CONTENTS

LIST OF TABLES

TABLE OF CONTENTS (CONTINUED)

LIST OF FIGURES

INTRODUCTION

Programs and Files

This report presents software and documentation for a brief practice simulation called "Storm" and for the Yugoslav Dilemma simulation used to assess participants' decision-making strategy. Software for the simulations is composed of eight programs:

1) TEDITOR (APPLE WRITER)
2) TEDIT
3) LEDIT
4) DEDIT
5) AEDIT
6) VEDIT
7) SIM
8) PROFILE (Formerly called MEASURE)

The TEDITOR (APPLE WRITER) program is a word processing program copyrighted by Apple Computer, Inc. which allows the user to create messages that will appear during the course of the simulation. The precise time during the simulation when each of these messages occurs is determined by the TEDIT program. The decision alternatives which can be selected by simulation participants are created by the DEDIT program. The LEDIT program defines the locations of movable objects in the scenario, and it also determines the scenario start time and the time multiplier. The AEDIT program performs a number of functions related to participants' decisions, while the VEDIT program keeps track of the location of all eight programs. The main simulation program, SIM, uses the output of the TEDITOR, TEDIT, LEDIT, DEDIT, AEDIT, and VEDIT programs to run the simulation. The PROFILE program is an analysis program which calculates 14 measures of participant performance. The measures are described in detail by Criswell, Unger, Swezey, and Streufert (1983)[1]. Figure 1 illustrates the relationships between the various programs.

[1] Criswell, E. L., Unger, K. W., Swezey, R. W., and Streufert, S. Researcher's Manual to Accompany the Yugoslav Dilemma (A Computer Simulation). (SAI Report No. SAI-84-02-178) McLean, VA: Science Applications, Inc., 1984.

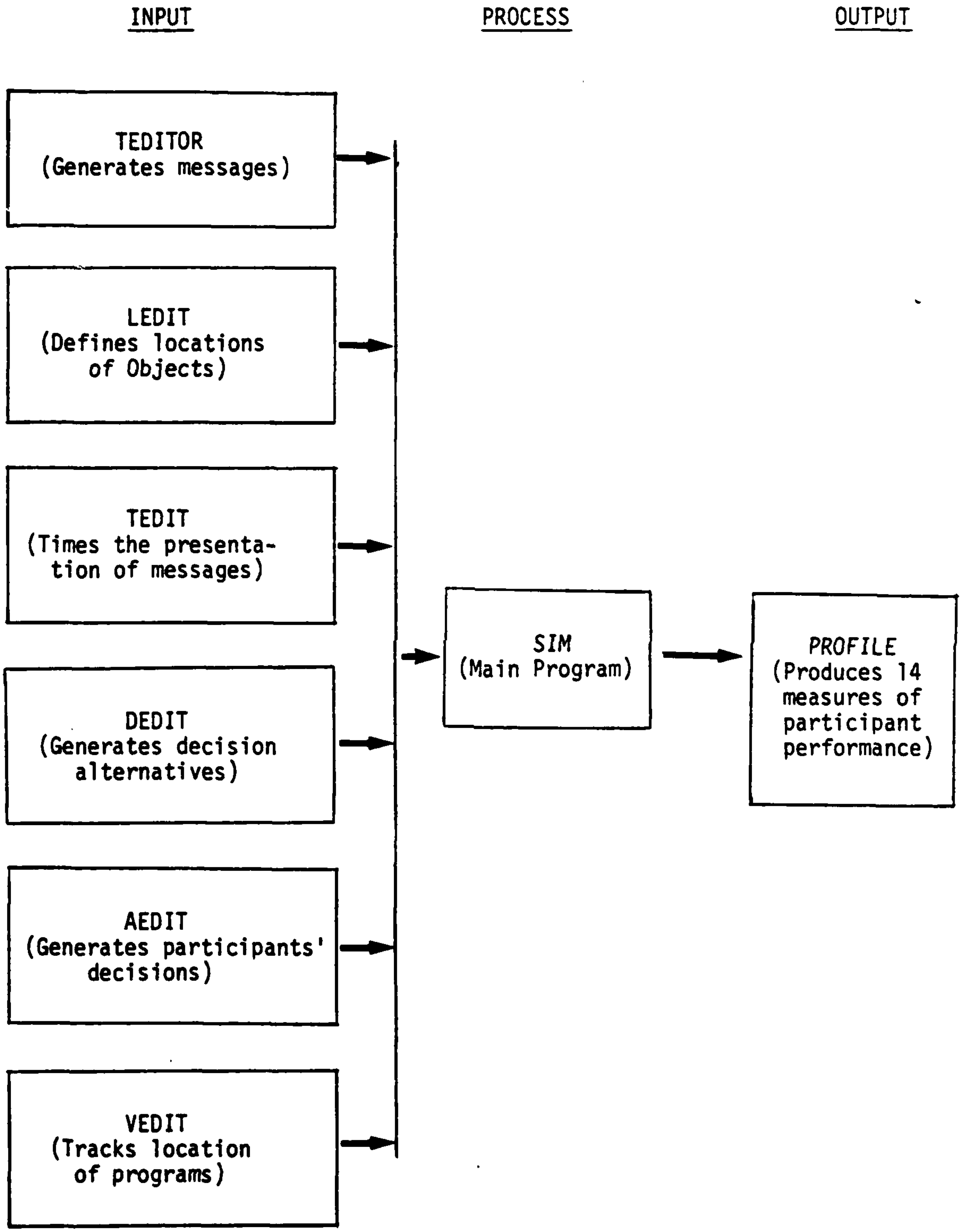

INPUT
PROCESS
OUTPUT
TEDITOR
(Generates messages)
LEDIT
(Defines locations of Objects)
TEDIT
(Times the presentation of messages)
DEDIT
(Generates decision alternatives)
AEDIT
(Generates participants' decisions)
VEDIT
(Tracks location of programs)
SIM
(Main Program)
PROFILE
(Produces 14 measures of participant performance)

Figure 1. System software

Each of the programs, except PROFILE, produces a file (or files) which is accessed during the simulation. TEDIT and SIM produce more than one file. PROFILE produces no files. The file(s) produced by each of the programs are as follows:

PROGRAM	FILE(S) PRODUCED
TEDITOR	TEXT. Mxxx
TEDIT	TM/SCENARIO NAME TS#/SCENARIO NAME
LEDIT	LOC/SCENARIO NAME
DEDIT	Drrrr
AEDIT	ATBL/SCENARIO NAME
VEDIT	V/SCENARIO NAME
SIM	R/PARTICIPANT NAME R#/PARTICIPANT NAME A/PARTICIPANT NAME

System Operation

A. Setup

Prior to beginning a simulation run, the system performs a series of checks and creates new files. The LOC/SCENARIO NAME file is checked to ensure that all objects have defined locations. Then a copy of the ATBL/SCENARIO NAME file is created and named A/PARTICIPANT NAME. This file keeps an updated record of changes in the simulation that are a function of time or a participant's decisions.

B. Message Presentation

After the setup has been completed and the screens containing instructions to the participant (TEXT.Mxxx files) have been presented, the SIM program reads the TM/SCENARIO NAME file to determine when messages should be presented and what messages to present. The messages are TEXT.Mxxx files. If no decisions are made, the simulation uses only these files during the course of the simulation.

C. Decision Making

When a participant presses the "D" key, the SIM program presents decision alternatives in the form of Drrrr files. The decisions which are executed, as well as future plans, and previous related decisions are stored on the R/PARTICIPANT NAME file. (This file should be cleared after each participant session. See Criswell et al., 1983, for details.)

D. Performance Measures

The PROFILE program reads the R/PARTICIPANT NAME file in order to compute the 14 performance measures.

Hardware Configuration

The hardware and operating manuals used to run the simulation are as follows:

1) Apple II Plus computer. The Apple computer is accompanied by the following manuals:

   a) Applesoft II Basic Programming Reference Manual - Provides in-depth explanations of all Applesoft commands.

   b) The Applesoft Tutorial - Introduces the user to programming techniques.

   c) Apple II Reference Manual - An accumulation of facts about Apple hardware.

2) Microsoft Ramcard and accompanying installation and operating instructions. This card is placed in the Apple's slot #0.

3) Apple II disk drives (2) and accompanying DOS Manual. The controller card for disk drive 1 is installed in the Apple's slot #6.

4) Thunderclock Plus clock card and accompanying installation and operating manual. The clock card is installed in slot #4.

5) Amdek Color I 13" monitor (no manuals).

6) Integral Data Systems 445G printer and accompanying owner's manual.

7) Grappler interface card and cable with accompanying operator's manual. The card is installed in slot #1.

8) Maezon 10 megabyte hard disk, controller card, and cable with accompanying installation and operating instructions. The controller card is installed in slot #5.

System Limitations

The limiting factor is the amount of core memory available. The 16K Ramcard is sufficient for current purposes; however, if significant amounts of code are added to the existing software, then a memory card or computer with greater capacity will be required.

TEDITOR PROGRAM

1.1 GENERAL INFORMATION

The TEDITOR (APPLE WRITER) program is used to generate messages which appear in the simulation. The following message numbers are reserved for specific uses:

M1-M400 - Used for fixed and responsive messages to be presented to the participant by SIM (e.g., Shortages of food are now common in Macedonia).

M401-M800 - Used for endings to be added to the end of a decision string chosen by a participant (e.g., was not successful).

M900-M999 (except M910 to M917) - Reserved for SIM program messages not normally modified (e.g., Are you planning any future decisions?).

M910-M917 - Reserved for eight messages to be used for successful endings for move verbs (e.g., has been successfully accomplished). If the researcher elects to have the participant receive only successful responses to his or her decisions, SIM program will randomly select one of these eight messages.

The actual file names that appear on disk are of the form TEXT.Mn (i.e., TEXT.M44).

1.2 USING TEDITOR

1.2.1 BRUN TEDITOR from the volume specified in the VEDIT program.

1.2.2 Generate messages using the normal APPLE WRITER commands. Wrapping words around the screen edge is not a problem since SIM fixes the line length.

1.2.3 Store the text on disk.

1.3 COMPLETE MESSAGES

1.3.1 Complete messages are generated with numbers from 1 to 400. Number assignments are arbitrary.

1.4 MESSAGE ENDINGS

1.4.1 Message endings for SIM decision strings are generated with numbers from M401 to M800.

1.4.2 Message endings must begin with the number 1, 2, or 3, indicating failure, neutral, or successful completion, respectively.

EXAMPLE: 1 was not successful.
3 has been completed.

1.4.3 The ending type number (1, 2, or 3) must agree with message type (if the type is 1, 2, or 3 from Section 2.4.4) when running TEDIT.

1.5 ACCOUNT ATTACHMENTS

An account is a location (file) with data. Account attachments direct data from one location to another.

1.5.1 Account attachments may be added to the end of messages (1.3 and 1.4) using TEDITOR. (This has not been done in the Yugoslav Dilemma; the Yugoslav Dilemma writes account attachments only in DEDIT.) This capability allows movement to occur in fixed messages.

1.5.2 Example: The flood stage has reached 12 feet @!4=>8@@!+5>9@ The attachment is @!4=>8@@!+5>9@.

1.5.3 Following from the definition of attachments in Section 4.3.7, the above example would substitute account 4 into account 8 and would add account 5 to account 9. Account 4 could be a flood stage account with the message "FLOOD STAGE IS 12 FEET." Account 5 could be a value of 2. Account 9 could be the number of failed levees. Therefore, this attachment would add 2 to the number of failed levees.

1.6 FILES

1.6.1 INPUT/OUTPUT FILES

NAME: TEXT.Mn
where n is the message number.
SOURCE: TEDITOR
EXAMPLE: TEXT.M33

1.7 ENDING PROGRAM

1.6.1 Write the output file onto the hard disk before leaving TEDITOR (APPLE WRITER).

1.8 PROGRAM LISTING

Since the APPLE WRITER (TEDITOR) program is a commercially available, copyrighted product of Apple, Inc., a complete listing with documentation is inappropriate for inclusion in this report.

TEDIT PROGRAM

2.1 GENERAL INFORMATION

The TEDIT program produces the time sequence files that are the heart of the scenario. These files determine how long the simulation runs, the times at which messages will appear (load), and the ratio of fixed to responsive messages.

2.2 PREREQUISITES

Prior to using TEDIT, message files must be generated with TEDITOR.

2.3 USING TEDIT

2.3.1 RUN TEDIT from the volume specified in the VEDIT program.

2.3.2 Command Menu:

```
E=Edit a file.
C=Cycle list of files.
P=Printer ON.
N=No Printer.
L=Load a scenario from disc.
S=Save scenario on disc.
Q=Quit.
?=Any other key gets above list.
```

2.3.3 To edit an existing scenario, first load (L) the scenario and then edit (E). To create a new scenario, simply use the E command.

2.4 PRODUCING A TIME LINE

2.4.1 Enter command E (Edit).

2.4.2 TEDIT will request a time with the message, ENTER T.

2.4.3 Enter the time that the message should appear, in integer real time minutes from 0 to 200. Two types of data must be defined for each time, a TSN (Time Status Number) and a MSL (Message Selection Line).

2.4.4 A TSN defines interpretation of the MSL according to the following table:

0=No message at this time. (Default value at start of TEDIT.)
1=Failure ending for a message in ml (file containing message 1) of MSL, with remainder being complete random messages in file m2 up to the nth message.
2=Neutral ending for a message in ml of MSL, with remainder being complete random messages in m2 to mn.
3=Success ending for a message in ml of MSL, with remainder in m2 to mn.
4=Random selection of one of the ml to mn messages in MSL with a check for redundancy.
5=Same as 4, above, except no check for redundancy.
6=Fixed message.
7=Take a break after displaying ml of TSN.
8=Do not allow "D" decision in SIM until a TSN of 9 (below) is found.
9=Allow "D" decision in SIM (reverses TSN of 8 above).

2.4.5 A MSL must be defined if the TSN is not zero. If a MSL has been previously defined, it will be displayed. The form of the MSL is:

ml; m2; m3; m4; ....; mn

where ml, etc. are message numbers.

For example: 441; 1, 5; 4
441 would be message file TEXT.M441,
1 would be message file TEXT.M1, etc.

NOTE: Message numbers indicate type of message, see paragraph 1.4.

2.4.6 TEDIT makes some checks to be sure that there is agreement between TSN and MSL. Primarily, this is a check of MSL against TSN. A warning (message string does not agree with message type, or message already exists) is produced and the problem can be corrected either by changing TSN or MSL. (The message can also be changed by TEDITOR.)

2.5 LISTING FILES

The files can be listed using the "C" command.

2.5.1 The data will be listed for the TSN and MSL files.

2.5.2 TEDIT will ask whether you want to see the matching messages from the TEXT.Mn files.

2.5.3 TEDIT will show the current first and last filled TSN file numbers. Select the range of time lines to be displayed.

2.6 FILES

2.6.1 INPUT FILES

NAME: TEXT.Mn
where n is 1-200, 401-800, 910-918.
SOURCE: TEDITOR

2.6.2 INPUT/OUTPUT FILES (each file is both input and output)

NAME: TS#/s
where s is scenario name
SOURCE: TEDIT
EXAMPLE: TS#/YUGOSLAV DILEMMA

NAME: TM/s
where s is scenario name
SOURCE: TEDIT
EXAMPLE: TM/YUGOSLAV DILEMMA

2.7 ENDING PROGRAM

2.7.1 The output files must be written using the "S" command when the files are correct. Writing interim files is a good practice to keep from losing all of the work in case of a fatal program error or a computer shutdown.

2.7.2 After selecting the "S" command, a scenario name must be entered. The output files will be written using that scenario name as an extension. For interim files, a scenario name such as "TEMP" could be used.

2.7.3 A warning will be issued if a Q (Quit) command is used and any changes have been made since the last "S" command.

2.8 PROGRAM LISTING

```
                                                          COMMENTS
-----------------------------------------------------------------------
                                              TEDIT PROGRAM START
1  REM  2150 1/30/83
10  REM  TEDIT
20 NN = 0: TEXT
30  HOME : VTAB 10: HTAB 10: PRINT
      "PROGRAM TEDIT": PRINT
50 D4$ =  CHR$ (4)
100  GOSUB 8000
-----------------------------------------------------------------------
                                              MENU COMMAND DISPLAY
110  HOME : PRINT "MENU"
200  PRINT "COMMAND (";: FOR I = 0 TO
       NA: PRINT  LEFT$ (C$(I),1);: NEXT
       I: PRINT "):";
210  GET A$: PRINT A$
212  FOR C = 0 TO NA: IF A$ =  LEFT$
       (C$(C),1) THEN 250
215  NEXT C
220  PRINT : PRINT : PRINT "COMMANDS:":
       FOR I = 0 TO NA: PRINT C$(I):
       NEXT I: GOTO 200
250  ON C + 1 GOTO
       300,600,1200,1500,1800,2100,3000,2
       400
-----------------------------------------------------------------------
                                              CODE FOR EDIT COMMAND
300  PRINT : PRINT "ENTER T:";: INPUT T
310  IF T < 0 OR T > 211 THEN  PRINT
       "ERROR, MUST BE 0 TO 211":
       GOTO 200
320  PRINT "TIME STATUS:"T%(T): PRINT
       "TIME MESSAGES:"T$(T)
325  PRINT :W = 1
330  FOR I = 0 TO 9: PRINT I"="TY$(I)"
       MESSAGE": NEXT I
340  PRINT : PRINT "ENTER 0 TO 9:";: GET
       N$: PRINT N$:N =  ASC (N$) - 48:
       IF N < 0 OR N > 9 THEN  PRINT
       "ERROR": GOTO 340
350 T%(T) = N: IF N THEN 400
360  IF T$(T) = "" THEN 599
370  FLASH : PRINT "WARNING, THE MESSAGE
       STRING IS NOT BLANK": NORMAL :
       PRINT "THIS IS NOT NECESSARILY
       BAD."
380  PRINT : PRINT "DO YOU WANT TO
       DELETE THE MESSAGE       STRING?
       (Y/N)";: GET A$: PRINT A$
390  IF A$ = "Y" THEN T$(T) = "": GOTO
       599
391  IF A$ <  > "N" THEN 370
400 M$ = T$(T): IF M$ = "" THEN 410
401  PRINT "CURRENT MESSAGE
       STRING="T$(T): PRINT
402  PRINT "DO YOU WANT TO CHANGE IT?
       (Y/N):";: GET A$: PRINT A$
403  IF A$ = "N" THEN 411
404  IF A$ <  > "Y" THEN 402
410  PRINT "ENTER MESSAGE STRING
       SEPARATING MESSAGE NUMBERS WITH
       SEMI-COLONS:";: INPUT M$
411  IF T > NN THEN NN = T
420  IF M$ = "" THEN 410
425  IF  LEN (M$) > 40 THEN  PRINT
       "ERR,40 CHARACTERS
       ALLOWED.": GOTO 401
```

```
430 QS$ = M$: GOSUB 9100
440  PRINT . PRINT "MESSAGE #'S:";: FOR
       I = 0 TO QN: PRINT QV(I)" ";:
       NEXT I
441  PRINT : POKE 16385,0
450  PRINT
       D4$"BLOADTEXT.M"QV(0)".A$4000"VS$(
       1): PRINT D4$"CLOSE":V =  PEEK
       (16385) - 240: IF V < 1 OR V > 9
       THEN V = 0
451  IF T%(T) > 3 THEN 460
452  IF V = T%(T) THEN 470
453  PRINT "ERROR. THE MESSAGE STRING
       DOES NOT AGREEWITH THE TYPE OF
       THE MESSAGE=";: PRINT T%(T)" IS
       NOT EQUAL TO "V: GOTO 400
460  IF V = 0 THEN 470
461  PRINT "ERROR. THE FIRST MESSAGE
       BEGINS WITH A  NUMBER AND THIS IS
       NOT LEGITIMATE EXCEPTFOR TYPES
       1.2, OR 3.";: GOTO 400
470  IF QN = 0 THEN 480
471  FOR I = 1 TO QN: PRINT
       D4$"BLOADTEXT.M"QV(I)",A$4000"VS$(
       1): PRINT D4$"CLOSE":V =  PEEK
       (16385) - 240: IF V < 1 OR V > 9
       THEN V = 0
472  IF V THEN  PRINT "ERROR, 2 THRU N
       MESSAGES MAY NOT BEGIN  WITH A
       NUMBER, MESSAGE #=";QV(I): GOTO
       400
480 T$(T) = M$
599  GOTO 200
-------------------------------------------------------------------
                                              CODE FOR CYCLE COMMAND
600  PRINT "CYCLE"
610  PRINT "DO YOU WANT TO SEE THE
       MESSAGES? (Y/N):": GET A$: PRINT
       A$
620  IF A$ = "Y" THEN S = 1: GOTO 623
621  IF A$ <  > "N" THEN 610
622 S = 0
623  FOR I = 0 TO NN: IF T%(I) THEN 625
624  NEXT I
625  PRINT "FIRST, LAST RECORDS="I","NN
630  INPUT "INPUT FIRST, LAST FOR
       CYCLE:";F,L
632  IF F > L OR (F < 0 OR L > 200) THEN
       PRINT "ERR": GOTO 630
634  PRINT "HIT ! TO STOP LISTING, ANY
       OTHER TO     STOP AND RESTART
       SCROLLING"
639  POKE  - 16368,0
640  FOR I = F TO L
641  INVERSE : PRINT
       "-------------------------------
       ----": NORMAL
650  PRINT I")"T%(I)":"TY$(T%(I))"
       MESSAGE"
652  IF  NOT T%(I) THEN 690
660 QS$ = T$(I): GOSUB 9100
670  FOR J = 0 TO QN: PRINT QV(J)
671  IF  NOT S THEN 680
672 QI = QV(J): GOSUB 9500
680  NEXT J
690  PRINT :X =  PEEK ( - 16384): IF X <
       128 THEN 699
691  POKE  - 16368,0
692  IF X = 161 THEN 200
```

```
693  GET A$
699  NEXT I
700  GOTO 200
------------------------------------------------------------------------------
                                                CODE FOR PRINTER ON/OFF COMMANDS
1200   PRINT D4$"PR#1": GOTO 200
1500   PRINT D4$"PR#0": GOTO 200
------------------------------------------------------------------------------
                                                CODE FOR LOAD COMMAND
1800   PRINT "LOAD ARRAY FROM DISC":
        PRINT
1810   INPUT "ENTER DATA SCENARIO
        NAME:";F$
1820   IF F$ = "" THEN 200
1825   PRINT D4$"VERIFY TM/"F$VS$(3)
1830   PRINT D4$"READTM/"F$
1840   INPUT NN
1841   PRINT "HIGHEST RECORD="NN
1850   FOR I = 0 TO NN: INPUT T%(I): NEXT
        I
1860   PRINT D4$"CLOSE"
1861   PRINT D4$"OPENTS#/"F$VS$(3)",L40"
1870   FOR I = 0 TO NN: IF  NOT T%(I)
        THEN 1900
1880   PRINT D4$"READTS#/"F$",R"I
1890   INPUT T$(I)
1900   NEXT I
1909   PRINT D4$"CLOSE"
1910   GOTO 200
------------------------------------------------------------------------------
                                                CODE FOR SAVE COMMAND
2100   PRINT "SAVE ARRAY ON DISC": PRINT
        :TT = NN
2101   FOR I = 0 TO NN:TN%(I) =
        T%(I):TN$(I) = T$(I): NEXT I:TT =
        NN
2102   IF F$ = "" THEN 2106
2103   PRINT "USE ("F$") FOR SCENARIO?
        (Y/N):";: GET A$: PRINT F$
2104   IF A$ = "Y" THEN 2109
2106   INPUT "ENTER SCENARIO NAME";F$
2109   GOSUB 2130: GOTO 200
2110   INPUT "ENTER SCENARIO NAME";F$
2120   IF F$ = "" THEN 2110
2130   PRINT D4$"OPENTM/"F$VS$(3)
2140   PRINT "HIGHEST RECORD ="TT
2141   INPUT "ENTER HIGHEST RECORD NUMBER
        TO BE      RECORDED:";NX
2142   PRINT D4$"WRITETM/"F$: PRINT NX
2150   FOR I = 0 TO NX: PRINT TN%(I):
        NEXT I
2160   PRINT D4$"CLOSE"
2161   PRINT D4$"OPENTS#/"F$VS$(3)",L40"
2162   PRINT D4$"CLOSE": PRINT
        D4$"DELETETS#/"F$VS$(3)
2163   PRINT D4$"OPENTS#/"F$VS$(3)",L40"
2170   FOR I = 0 TO NX: IF  NOT TN%(I)
        THEN 2200
2180   PRINT D4$"WRITETS#/"F$",R"I
2190   PRINT TN$(I)
2200   NEXT I
2205 W = 0
2209   PRINT D4$"CLOSE"
2210   RETURN
------------------------------------------------------------------------------
                                                CODE FOR QUIT COMMAND
2400   IF  NOT W THEN  END
2410   FLASH : PRINT "WARNING,
        RECORDS NOT WRITTEN ON DISC,   IF
```

```
       THIS IS OK HIT AN ASTERISK (*),
       ANY OTHER KEY TO RETURN TO
       MENU.": NORMAL : GET A$: IF A$ <
       > "*" THEN 200
2420  END
--------------------------------------------------------------------------
                                                 CODE FOR BUILD COMMAND
3000  REM
3001  FOR I = 0 TO 211:TN%(I) = 0:TN$(I)
       = "": NEXT I:I = FRE (0)
3005  INPUT "ENTER NUMBER OF
       PERIODS:";NP%: IF NP% < 1 OR NP%
       > 9 THEN PRINT "ERR-1 TO 9":
       GOTO 3005
3006  INPUT "ENTER GAP MINUTES (0 OR
       MORE):";GP%: IF GP% < 0 THEN
       PRINT "ERR": GOTO 3006
3007  IF GP% THEN PRINT "ENTER LOCKOUT
       MINUTES ("GP%" OR LESS):";: INPUT
       LM%: IF LM% > GP% THEN PRINT
       "ERR": GOTO 3007
3010  PRINT "ENTER MESSAGES, MINUTES FOR
       EACH PERIOD"
3011 SM% = 0
3020  FOR I = 1 TO NP%
3021  PRINT "ENTER FOR PERIOD="I":",:
       INPUT MP%(I),TP%(I)
3025 AM%(I) = TP%(I) - GP%
3030  IF AM%(I) < MP%(I) THEN PRINT
       "ERROR, WON'T FIT": GOTO 3021
3035  IF NOT MP%(I) THEN PRINT
       "ERROR-NUMBER OF MESSAGES
       LESS THAN 1": GOTO 3021
3040 SM% = SM% + MP%(I)
3060  NEXT I:SM% = SM% - 1
3070  IF SM% > NN THEN PRINT
       "ERROR-YOU REQUESTED "SM%"
       MESSAGES": PRINT "YOU ONLY HAVE
       "NN" AVAILABLE": GOTO 200
3080  FOR I = 0 TO SM%: IF NOT T%(I)
       THEN PRINT "ERROR-NO MESSAGE
       FOUND AT "I:J = 1
3090  NEXT I: IF J THEN 200
3100 SM% = 0:TT = 0:MC% = 0
3110  FOR I = 1 TO NP%:MP% = MP%(I):AM%
       = AM%(I)
3120  GOSUB 3900
3130  IF NOT MP% THEN 3200
3140 K = ((AM% - MP%) / MP%) + .000001
3160 TT = TT + K: GOSUB 3900: IF MP%
       THEN 3160
3200 TT = SM% + TP%(I)
3205  IF LM% THEN TN%(TT - LM%) =
       8:TN$(TT - LM%) = "80"
3210 TN%(TT) = 7:TN$(TT) = "70"
3212  IF I = NP% THEN TN$(TT) = "71"
3215 TT = TT + 1
3220 SM% = TT: NEXT I
3230 TT = TT - 1
3240  FLASH : PRINT "SAVING NEW ARRAY TO
       DISC": NORMAL
3250  GOSUB 2110
3260  GOTO 200
3900 TN%(TT) = T%(MC%):TN$(TT) = T$(MC%)
3910 MC% = MC% + 1:TT = TT + 1:AM% = AM%
       - 1:MP% = MP% - 1
3920  RETURN
7999  END
--------------------------------------------------------------------------
```

SETUP SUBROUTINE

```
------------------------------------------------------------------------
                                                DIMENSION VARIABLES
8000  DIM
       T%(211),T$(211),VS$(16),TN%(211),T
       N$(211)
------------------------------------------------------------------------
                                                INITIALIZE VARIABLES
8010  FOR NA = 0 TO 999: READ C$(NA)
8020  IF C$(NA) = "$" THEN 8041
8030  DATA  EDIT,CYCLE,PRINTER ON,NO
       PRINTER,LOAD FROM DISC,SAVE TO
       DISC,BUILD A SCENARIO,QUIT
8039  DATA $
8040  NEXT NA
------------------------------------------------------------------------
                                                READ V/SCENARIO FILE
8041 NA = NA - 1
8042  INPUT "ENTER VOLUME SCENARIO:";X$:
       PRINT D4$"VERIFYV/"X$: PRINT
       D4$"READV/"X$: FOR I = 0 TO 16
8043  GET A$: IF  ASC (A$) = 13 THEN
       8045
8044 VS$(I) = VS$(I) + A$: GOTO 8043
8045  NEXT I: PRINT D4$: PRINT
       D4$"CLOSE"
8060  FOR NT = 0 TO 9: READ TY$(NT)
8071  DATA
       NO,FAILURE,NEUTRAL,SUCCESS,RANDOM
       (NOT CHECKED),RANDOM (REDUNDANT
       CHECK),UNUSED, BREAK,LOCKOUT
       'D',ALLOW 'D'
8075  NEXT NT
8999  RETURN
------------------------------------------------------------------------
                                                SEARCHES A STRING FOR "#",
                                                ";", AND "&" SYMBOLS
9100 QW = 0:QO = 0:QN = 0:QL =  LEN
       (QS$):QV(1) = 0:QE = QL:QV(0) =
       VAL (QS$): FOR QI = 2 TO QL: IF
       MID$ (QS$,QI,1) (   ) ";" THEN
       9130
9110 QN = QN + 1:QI = QI + 1:QV(QN + 1)
       = 0:QV(QN) =  VAL ( MID$
       (QS$,QI,99))
9120  GOTO 9190
9130  IF  MID$ (QS$,QI,1) (   ) "#" THEN
       9160
9140 QI = QI + 1:QO =  VAL ( MID$
       (QS$,QI,99))
9150  GOTO 9190
9160  IF  MID$ (QS$,QI,1) (   ) "&" THEN
       9199
9170 QI = QI + 1:QW = 1:QW$ =  MID$
       (QS$,QI,99)
9190  IF QE ) QI - 2 THEN QE = QI - 2
9199  NEXT QI: RETURN
------------------------------------------------------------------------
                                                DECODES X,Y CHARACTERS
                                                TO NUMBERS
9200 X = ( ASC ( LEFT$ (QS$,1)) - 65) *
       26 +  ASC ( MID$ (QS$,2,1)) -
       65:Y =  VAL ( MID$ (QS$,3,99))
9210  RETURN
9300 QD(QN) = DT(QN) + T
9310  FOR QB = QN TO 4:QI =  INT
       ((QD(QB) - (QB ) 1)) / QS(QB))
9320 QD(QB) = QD(QB) - QI * QS(QB):QD(QB
       + 1) = DT(QB + 1) + QI: NEXT QB
```

```
9330  RETURN
--------------------------------------------------------------------------
                                                    READS AN M900 FILE
9399 D4$ =  CHR$ (4):QI = 900: PRINT
       D4$"VERIFYM"QI;VS$(1)
9400  PRINT D4$"READM"QI
9410 A$ = ""
9412  GET QQ$: IF QQ$ =  CHR$ (13) THEN
        9415
9413 A$ = A$ + QQ$: GOTO 9412
9415  IF  LEFT$ (A$,1) = "]" THEN  PRINT
        : GOTO 9499
9420 QB = 1:QN =  VAL (A$): IF QN THEN
       QB = 2
9422  IF QN < 0 THEN QB = 3:QN =  - QN:
        PRINT
9423  PRINT
9425  IF QN = 1 THEN  INVERSE
9426  IF QN = 2 THEN  FLASH
9430  PRINT  MID$ (A$,QB,255);
9440  NORMAL : GOTO 9410
9499  PRINT D4$"CLOSE": RETURN
--------------------------------------------------------------------------
                                                    READS A BINARY TEXT.Mxxx
                                                    FILE AND CONVERTS IT TO
                                                    APPLESOFT CHARACTERS.
9500  PRINT
       D4$"BLOADTEXT.M"QI",A$4000"VS$(1):
       QL =  PEEK (43616) +  PEEK
       (43617) * 256 - 1
9505 QB = 1
9510 QN = QB + 39: IF QN > QL THEN QN =
       QL
9515 QQ = QI
9516  IF  PEEK (16384 + QB) = 141 THEN
        PRINT :QB = QB + 1: GOTO 9510
9520  FOR QI = QB TO QN:QV =  PEEK (QI +
        16384)
9530  IF QV = 141 THEN QQ = QI: GOTO
        9550
9540  IF (QV = 224) OR ((QV = 32) OR (QV
        = 96)) THEN QQ = QI - 1
9549  NEXT QI
9550  FOR QJ = QB TO QQ:QA =  PEEK (QJ +
        16384): IF QA < 64 THEN  INVERSE
        :QA = QA + 64
9551  IF QA > 223 THEN QA = QA - 64
9553  PRINT  CHR$ (QA);: NORMAL : NEXT
        QJ:QB = QQ + 2
9554 QV =  PEEK (16384 + QB): IF (QV =
        224) OR ((QV = 32) OR (QV = 96))
        THEN QB = QB + 1: GOTO 9554
9555  IF QB < QL THEN  PRINT : GOTO 9510
9570  RETURN
9800 QB = 1:QS$(0) = QS$:QN = 0:QL =
        LEN (QS$): IF QL < 40 THEN
        RETURN
9805 QN =  - 1
9820  FOR QI = QB + 38 TO QB STEP  - 1
9830  IF  MID$ (QS$,QI,1) = " " THEN
        9850
9840  NEXT QI: PRINT "END9840": END
9850 QN = QN + 1:QS$(QN) =  MID$
        (QS$,QB,QI - QB + 1):QB = QI + 1
9855  IF QL - QB > 39 THEN 9820
9856 QN = QN + 1:QS$(QN) =  RIGHT$
        (QS$,QL - QB + 1)
9860  RETURN
--------------------------------------------------------------------------
```

UTILITY ENTRIES NOT
EXECUTED BY TEDIT

```
30000  INPUT B$: PRINT B$
50000 I$ =  CHR$ (9):Q$ =  CHR$ (27):D$
       =  CHR$ (4):S$ =  CHR$ (31):M$ =
       CHR$ (30):L$ =  CHR$ (29):NC$ =
       CHR$ (2):EX$ =  CHR$ (1)
50002  PRINT D$"PR#0"
50005  PRINT D$"PR#1"
50006  PRINT Q$"J.0,960,$"
50007  PRINT Q$"B,6,$"
50010  PRINT I$0"N"
50020  PRINT Q$"R,2,$"
50030  PRINT M$NC$
50100  END
55000 D$ =  CHR$ (4): PRINT D$"OPEN
       ADDLIST": PRINT D$"WRITE
       ADDLIST": LIST : PRINT D$"CLOSE":
       END
```

KEY VARIABLES

T = MESSAGE SLOT TO BE EDITED
T$ = MESSAGE STRING IN MESSAGE SLOT CURRENTLY ACCESSED
NN = NUMBER OF MESSAGE SLOTS IN SCENARIO
F = FIRST MESSAGE SLOT TO BE REVIEWED WHEN CYCLING
L = LAST MESSAGE SLOT TO BE REVIEWED WHEN CYCLING
F$ = SCENARIO NAME

LEDIT PROGRAM

3.1 GENERAL INFORMATION

The LEDIT program produces the files that locate the various (moveable and unmoveable) objects in a scenario and define their types. The program also produces the start time for the scenario, the time multiplier which sets the ratio of real to simulation time, and the charge time which sets the amount of time charged for each decision.

3.2 PREREQUISITES

LEDIT requires that all of the objects be located on a map.

3.3 USING LEDIT

3.3.1 RUN LEDIT from the volume specified in VEDIT.

3.3.2 Command Menu:

E=Edit a location record. (See 3.4)
C=Cycle list of location records.
P=Printer ON.
N=No Printer.
L=Load a scenario from disc.
S=Save scenario on disc.
T=Time and date edit. (See 3.5)
Q=Quit.
?=Any other key gets above list.

3.3.3 To edit an existing scenario, first load (L) the scenario and then edit (E). To create a new scenario, simply use the E command.

3.4 PRODUCING A LOCATION RECORD

3.4.1 Enter command E (Edit).

3.4.2 LEDIT will request an object number with the message, ENTER OBJECT NUMBER:
(Selections N, S, X, Y, Q, C, M are then produced. See 3.4.3 through 3.4.9.)

3.4.3 N=Enter a DESCRIPTIVE NAME. Enter an "N" and then the name of the object. This name is for information purposes only and is not used in subsequent programs. However, it is necessary that a name be entered, as this is how LEDIT determines whether to pass location data on to SIM.

3.4.4 S=STATUS NUMBER. Either 0 or 1 must be entered for the object.

0=Unmoveable object.
1=Moveable object.

3.4.5 X=Define an integer x coordinate for the object. A value of less than 1 is not allowed. *Instead of* x, y coordinates, a Quadrant is allowed. (See 3.4.7.)

3.4.6 Y=Define an integer y coordinate for the object. A value of less than 1 is not allowed.

3.4.7 Q=Quadrant definition. An object can either be defined with a quadrant definition or an x, y definition. Quadrants run from AA to ZZ on the horizontal axis and from 1 to infinity on the vertical axis.

3.4.8 C=CANCEL that object.

3.4.9 M=Return to the COMMAND MENU (3.3.2)

3.4.10 NOTE: The origin of the Quadrant record is defined as OBJECT 0 (zero). You must define the origin before writing a scenario to disk even if you do not intend to use quadrants.

3.5 PRODUCING A TIME RECORD

3.5.1 The TIME Record is used to set the displayed date at the start of the SIM program, the time multiplier for the display time, and the time charged for making a decision. Press the T key to observe these values.

3.5.2 START DATE AND TIME. These values are easily created or modified by entering new values on the keyboard.

3.5.3 The TIME MULTIPLIER is the number of seconds of DISPLAY TIME that pass for each second of REAL TIME. For example, a 60 would cause DISPLAY TIME to change by 60 seconds for each second of REAL TIME; 120, 2 minutes for every two seconds, or 1 hour for every 30 seconds.

3.5.4 The CHARGE TIME multiplied by the TIME MULTIPLIER yields the number of REAL TIME seconds charged for the decision loop in SIM. For example, if the CHARGE TIME is 10 and the TIME MULTIPLIER is 60, then 10 times 60 seconds (10 minutes) of DISPLAY TIME would pass during a decision. If the CHARGE TIME is 30 and the TIME MULTIPLIER is 120, one hour is charged for each decision.

3.6 LISTING FILES

The OBJECT records can be listed using the "C" command.

3.6.1 The "C" command only lists the OBJECT Records. The Time Record is listed using the "T" command.

3.6.2 LEDIT will show the current first and last filled OBJECT Record numbers. Select the range of time lines to be displayed.

3.7 FILES

3.7.1 INPUT/OUTPUT FILES

NAME: LOC/s
where s is scenario name
SOURCE: LEDIT
EXAMPLE: LOC/YUGOSLAV DILEMMA

3.8 ENDING PROGRAM

3.8.1 The output files must be written using the "S" command when the files are correct. Writing interim files is a good practice to keep from losing all of the work in case of a fatal error or a computer shutdown.

3.8.2 After selecting the "S" command, a scenario name must be entered. The output files will be written using that scenario name as an extension. For interim files, a scenario name such as "TEMP" could be used.

3.8.3 If any changes have been made since the last "S" command and a "Q" command is entered, a warning will be issued.

3.9 PROGRAM LISTING

LEDIT

```
                                                     COMMENTS
-------------------------------------------------------------------------
                                                START LEDIT PROGRAM
10  HOME : PRINT "LEDIT PROGRAM"
11  TEXT
20 NN = 0
50 D4$ =  CHR$ (4)
100  GOSUB 8000
-------------------------------------------------------------------------
                                                COMMANDS THAT APPEAR IN
                                                THE MENU
110  VTAB 2: PRINT "MENU"
200  PRINT "COMMAND (";: FOR I = 0 TO
       NA: PRINT  LEFT$ (C$(I),1);: NEXT
       I: PRINT "):";
210  GET A$: PRINT A$
212  FOR C = 0 TO NA: IF A$ =  LEFT$
       (C$(C),1) THEN 250
215  NEXT C
220  PRINT : PRINT : PRINT "COMMANDS:":
       FOR I = 0 TO NA: PRINT C$(I):
       NEXT I: GOTO 200
250  ON C + 1 GOTO
       300,600,1200,1500,1800,2100,2700,2
       400
-------------------------------------------------------------------------
                                                CODE FOR THE EDIT COMMAND
300  HOME : VTAB 20: INPUT "ENTER OBJECT
       NUMBER:";T
310  IF T < 0 OR T > 100 THEN  PRINT
       "ERROR, MUST BE 0 TO 100":
       GOTO 200
315  HOME
320  PRINT "   NAME:"N$(T): PRINT : FOR
       J = 0 TO 2
322  PRINT TY$(J)":";
323  IF J = 1 AND OM%(1,T) < 0 THEN
       HTAB 1: PRINT
       "QUADRANT:";Q$(T);:J = J + 1:
       GOTO 330
324  PRINT OM%(J,T);
329  IF  NOT J THEN  PRINT "
       "S$(OM%(J,T))
330  PRINT : PRINT : NEXT J
339  VTAB 20: HTAB 1: PRINT
340  VTAB 20: HTAB 1: PRINT "SELECT (";:
       FOR J = 0 TO 6: PRINT  LEFT$
       (G$(J),1);: NEXT J: PRINT "):";
350  GET A$: PRINT A$
360  FOR J = 0 TO 6: IF  LEFT$ (G$(J),1)
       = A$ THEN 370
365  NEXT J: PRINT ""
366  VTAB 18: HTAB 1: FOR J = 0 TO 6:
       PRINT G$(J)",";: NEXT J: GOTO 339
370  VTAB 18: HTAB 1: PRINT : PRINT :
       PRINT : VTAB 20
390  ON J + 1 GOTO
       410,420,430,440,450,460,470
410  INPUT "ENTER DESCRIPTIVE
       NAME:";N$(T)
411 W = 1
412  GOTO 315
420  PRINT "ENTER STATUS NUMBER (0 OR
       1):";: GET A$: PRINT A$:A =  ASC
       (A$) - 48
421  IF A < 0 OR A > 1 THEN  PRINT "":
       GOTO 420
422 OM%(0,T) = A:W = 1: GOTO 315
```

```
430  PRINT "                          ";:
       HTAB 1: INPUT "ENTER
       X:";OM%(1,T):Q$(T) = ""
431 W = 1
436  GOTO 315
440  PRINT "
       ";: HTAB 1: INPUT "ENTER
       Y:";OM%(2,T):Q$(T) = ""
441 W = 1: GOTO 315
450  HTAB 1: PRINT "
       ";: HTAB 1: INPUT "ENTER
       QUADRANT:";QS$
451 W = 1: GOSUB 9200: IF X < 0 OR X >
       700 THEN  PRINT "ERROR": GOTO
       315
452  IF Y < 1 OR Y > 999 THEN  PRINT
       "ERROR": GOTO 315
453 Q$(T) = QS$:OM%(1,T) =  - 1: GOTO
       315
460 OM%(1,T) = 0:OM%(2,T) = 0:Q$(T) =
       "":N$(T) = "":OM%(0,T) = 0: GOTO
       315
470  IF OM%(1,T) = 0 AND N$(T) <  > ""
       THEN  PRINT "ERROR, NAME WITH
       X=0, HIT KEY TO GO": GET A$: GOTO
       315
471  IF OM%(1,T) <  > 0 AND N$(T) = ""
       THEN  PRINT "ERROR, X=0 WITH
       NAME, HIT KEY TO GO": GET A$:
       GOTO 315
472  GOTO 110
599  GOTO 200
-----------------------------------------------------------------------
                                             CODE FOR THE CYCLE COMMAND
600  PRINT "CYCLE"
630  INPUT "INPUT FIRST, LAST FOR
       CYCLE:";F,L
632  IF F > L OR (F < 0 OR L > 200) THEN
       PRINT "ERR": GOTO 630
634  PRINT "HIT ! TO STOP LISTING, ANY
       OTHER TO     STOP AND RESTART
       SCROLLING"
639  POKE  - 16368,0
640  FOR I = F TO L
641  INVERSE : PRINT
       "---------------------------------
       ----": NORMAL
650  PRINT I")"N$(I)
660  PRINT : FOR J = 0 TO 2
662  PRINT TY$(J)":";
664  IF J = 1 AND OM%(1,I) < 0 THEN
       HTAB 1: PRINT
       "QUADRANT:";Q$(I);:J = J + 1:
       GOTO 680
666  PRINT OM%(J,I);
668  IF  NOT J THEN  PRINT "
       "S$(OM%(J,I))
680  PRINT : PRINT
682  NEXT J
683  IF N$(I) <  > "" THEN  FOR QQ = 0
       TO 1000: NEXT QQ
690 X =  PEEK ( - 16384): IF X < 128
       THEN 699
691  POKE  - 16368,0
692  IF X = 161 THEN 200
693  GET A$
699  NEXT I
700  GOTO 200
-----------------------------------------------------------------------
```

```
                                             PRINTER ON/OFF COMMANDS
1200  PRINT D4$"PR#1": GOTO 200
1500  PRINT D4$"PR#0": GOTO 200
-----------------------------------------------------------------------
                                             CODE FOR LOAD COMMAND
1800  PRINT "LOAD ARRAY FROM DISC":
       PRINT
1810  INPUT "ENTER SCENARIO NAME:";F$
1820  IF F$ = "" THEN 200
1829  PRINT D4$"VERIFYLOC/"F$",D1"
1830  PRINT D4$"READLOC/"F$
1841 NN = 100
1845  FOR I = 0 TO 7: INPUT T(I): NEXT I
1850  FOR I = 0 TO NN: FOR J = 0 TO 2:
       INPUT OM%(J,I): NEXT J: NEXT I
1855  FOR I = 0 TO NN: IF OM%(1,I) < 0
       THEN  INPUT Q$(I)
1856  NEXT I
1857  FOR I = 0 TO NN: IF OM%(1,I) <  >
       0 THEN  INPUT N$(I)
1858  NEXT I
1860  PRINT D4$"CLOSE"
1910  GOTO 200
-----------------------------------------------------------------------
                                             CODE FOR SAVE COMMAND
2100  PRINT "SAVE ARRAY ON DISC": PRINT
2101  IF  NOT OM%(1,0) OR  NOT OM%(2,0)
       THEN  PRINT "ERROR: ORIGIN
       UNDEFINED. HIT ANY KEY TO
       CONTINUE": GET A$: GOTO 200
2102  IF F$ = "" THEN 2110
2103  PRINT "USE ("F$") FOR SCENARIO?
       (Y/N):";: GET A$: PRINT F$
2104  IF A$ = "Y" THEN 2130
2110  INPUT "ENTER SCENARIO NAME:";F$
2120  IF F$ = "" THEN 200
2130  PRINT D4$"OPENLOC/"F$
2142  PRINT D4$"WRITELOC/"F$
2145  FOR I = 0 TO 7: PRINT T(I): NEXT I
2150  FOR I = 0 TO 100: FOR J = 0 TO 2:
       PRINT OM%(J,I): NEXT J: NEXT I
2155  FOR I = 0 TO 100: IF OM%(1,I) < 0
       THEN  PRINT Q$(I)
2156  NEXT I
2157  FOR I = 0 TO 100: IF OM%(1,I) <  >
       0 THEN  PRINT N$(I)
2158  NEXT I
2160  PRINT D4$"CLOSE"
2205 W = 0
2210  GOTO 200
-----------------------------------------------------------------------
                                             CODE FOR QUIT COMMAND
2400  IF  NOT W THEN  END
2410  FLASH : PRINT "WARNING, FILES NOT
       WRITTEN ON DISC,     IF THIS IS
       OK HIT AN ASTERISK (*),      ANY
       OTHER KEY TO RETURN TO MENU.":
       NORMAL : GET A$: IF A$ <  > "*"
       THEN 200
2420  END
-----------------------------------------------------------------------
                                             CODE FOR TIME COMMAND
2700  HOME : PRINT "STARTING TIME.
       MULTIPLIER AND TIME CHARGE":
       PRINT : PRINT : FOR I = 0 TO 7:
       VTAB I + 4: HTAB 1: PRINT
       T$(I)"="T(I): NEXT I
2710  VTAB 20: HTAB 1: PRINT "OK
       (Y/N):";: GET A$: PRINT A$: IF A$
```

```
         = "Y" THEN 200
2711   IF A$ <   > "N" THEN 2710
2720   VTAB 20: HTAB 1: PRINT "
         ":: HTAB 1: PRINT "ENTER VALUE OR
         HIT RETURN FOR NO CHANGE"
2730   FOR I = 0 TO 7: VTAB I + 4: HTAB
         4: INPUT A$
2740   IF A$ = "" THEN 2760
2750 T(I) =  VAL (A$)
2760   VTAB I + 4: HTAB 1: PRINT
         T$(I)"="T(I)
2790   NEXT I: GOTO 2700
7999   END
--------------------------------------------------------------------------
                             SETUP SUBROUTINE
--------------------------------------------------------------------------
                                              DIMENSION VARIABLES
8000   DIM N$(100),OM%(8,100),Q$(100)
--------------------------------------------------------------------------
                                              INITIALIZE VARIABLES
8001   FOR NA = 0 TO 6: READ C$(NA): NEXT
         NA: DATA
         NAME,STATUS,X,Y,QUADRANT,CANCEL,ME
         NU
8010   FOR NA = 0 TO 999: READ C$(NA)
8020   IF C$(NA) = "$" THEN 8050
8030   DATA EDIT,CYCLE,PRINTER ON,NO
         PRINTER,LOAD FROM DISC,SAVE TO
         DISC,TIME,QUIT
8039   DATA $
8040   NEXT NA
8050 NA = NA - 1
8052 S$(0) = ">NON-MOVING LOC.":S$(1) =
         ">MOVABLE LOC.":S$(2) = ">MOVING"
8060   FOR NT = 0 TO 8: READ TY$(NT)
8071   DATA
         "STATUS",X,Y,SS,MM,HH,DD,MO,YR
8075   NEXT NT
8100   FOR I = 0 TO 7: READ T$(I): NEXT
         I: DATA SEC,MIN,HRS,DAY,MON,"
         YR",MUL,"CHR"
8999   RETURN
--------------------------------------------------------------------------
                                              SEARCHES A STRING FOR
                                              SPECIAL IDENTIFIERS
9100 QW = 0:QO = 0:QN = 0:QL =  LEN
         (QS$):QV(1) = 0:QE = QL:QV(0) =
         VAL (QS$): FOR QI = 2 TO QL: IF
         MID$ (QS$,QI,1) <   > ";" THEN
         9130
9110 QN = QN + 1:QI = QI + 1:QV(QN + 1)
         = 0:QV(QN) =  VAL ( MID$
         (QS$,QI,99))
9120   GOTO 9190
9130   IF  MID$ (QS$,QI,1) <   > "#" THEN
         9160
9140 QI = QI + 1:QO =  VAL ( MID$
         (QS$,QI,99))
9150   GOTO 9190
9160   IF  MID$ (QS$,QI,1) <   > "&" THEN
         9199
9170 QI = QI + 1:QW = 1:QW$ =  MID$
         (QS$,QI,99)
9190   IF QE > QI - 2 THEN QE = QI - 2
9199   NEXT QI: RETURN
9200 X = ( ASC ( LEFT$ (QS$,1)) - 65) *
         26 +  ASC ( MID$ (QS$,2,1)) -
         65:Y =  VAL ( MID$ (QS$,3,99))
9210   RETURN
```

```
9300 QD(QN) = DT(QN) + T
9310  FOR QB = QN TO 4:QI =  INT
        ((QD(QB) - (QB > 1)) / QS(QB))
9320 QD(QB) = QD(QB) - QI * QS(QB):QD(QB
        + 1) = DT(QB + 1) + QI: NEXT QB
9330  RETURN
----------------------------------------------------------------------------
                                                         READS MESSAGE STRINGS
9399 D4$ =  CHR$ (4):QI = 900
9400  PRINT D4$"READM"QI
9410 A$ = ""
9412  GET QQ$: IF QQ$ =  CHR$ (13) THEN
        9415
9413 A$ = A$ + QQ$: GOTO 9412
9415  IF  LEFT$ (A$,1) = "]" THEN  PRINT
        : GOTO 9499
9420 QB = 1:QN =  VAL (A$): IF QN THEN
        QB = 2
9422  IF QN < 0 THEN QB = 3:QN =  - QN:
        PRINT
9423  PRINT
9425  IF QN = 1 THEN  INVERSE
9426  IF QN = 2 THEN  FLASH
9430  PRINT  MID$ (A$,QB,255);
9440  NORMAL : GOTO 9410
9499  PRINT D4$"CLOSE": RETURN
----------------------------------------------------------------------------
                                                         CONVERTS BINARY FILES FROM
                                                         APPLEWRITER INTO APPLESOFT
                                                         CHARACTER STRINGS.
9500  PRINT
        D4$"BLOADTEXT.M"QI",A$4000":QL =
        PEEK (43616) +  PEEK (43617) *
        256 - 1
9505 QB = 1
9510 QN = QB + 39: IF QN > QL THEN QN =
        QL
9515 QQ = QI
9516  IF  PEEK (16384 + QB) = 141 THEN
        PRINT :QB = QB + 1: GOTO 9510
9520  FOR QI = QB TO QN:QV =  PEEK (QI +
        16384)
9530  IF QV = 141 THEN QQ = QI: GOTO
        9550
9540  IF (QV = 224) OR ((QV = 32) OR (QV
        = 96)) THEN QQ = QI - 1
9549  NEXT QI
9550  FOR QJ = QB TO QQ:QA =  PEEK (QJ +
        16384): IF QA < 64 THEN  INVERSE
        :QA = QA + 64
9551  IF QA > 223 THEN QA = QA - 64
9553  PRINT  CHR$ (QA);: NORMAL : NEXT
        QJ:QB = QQ + 2
9554 QV =  PEEK (16384 + QB): IF (QV =
        224) OR ((QV = 32) OR (QV = 96))
        THEN QB = QB + 1: GOTO 9554
9555  IF QB < QL THEN  PRINT : GOTO 9510
9570  RETURN
----------------------------------------------------------------------------
                                                         SPLITS A LINE INTO 40
                                                         CHARACTER STRINGS WITHOUT
                                                         BREAKING WORDS.
9800 QB = 1:QS$(0) = QS$:QN = 0:QL =
        LEN (QS$): IF QL < 40 THEN
        RETURN
9805 QN =  - 1
9820  FOR QI = QB + 38 TO QB STEP  - 1
9830  IF  MID$ (QS$,QI,1) = " " THEN
        9850
```

```
9840  NEXT QI: PRINT "END9840": END
9850 QN = QN + 1:QS$(QN) =  MID$
      (QS$,QB,QI - QB + 1):QB = QI + 1
9855  IF QL - QB ) 39 THEN 9820
9856 QN = QN + 1:QS$(QN) =  RIGHT$
      (QS$,QL - QB + 1)
9860  RETURN
------------------------------------------------------------------
                                         UTILITY ENTRIES NOT
                                         EXECUTED BY LEDIT
30000  INPUT B$: PRINT B$
50000 I$ =  CHR$ (9):Q$ =  CHR$ (27):D$
       =  CHR$ (4):S$ =  CHR$ (31):M$ =
       CHR$ (30):L$ =  CHR$ (29):NC$ =
       CHR$ (2):EX$ =  CHR$ (1)
50002  PRINT D$"PR#0"
50005  PRINT D$"PR#1"
50006  PRINT Q$"J,0,960,$"
50007  PRINT Q$"B,6,$"
50010  PRINT I$0"N"
50020  PRINT Q$"R,2,$"
50030  PRINT M$NC$
50100  END
55000 D$ =  CHR$ (4): PRINT D$"OPEN
       ADDLIST": PRINT D$"WRITE
       ADDLIST": LIST : PRINT D$"CLOSE":
       END
```

KEY VARIABLES

T	= OBJECT NUMBER TO BE CREATED OR EDITED
N$(T)	= NAME OF OBJECT TO BE CREATED OR EDITED
OM%(1,T)	= X COORDINATE OF OBJECT TO BE CREATED OR EDITED
OM%(2,T)	= Y COORDINATE OF OBJECT TO BE CREATED OR EDITED
QS$	= QUADRANT OF OBJECT TO BE CREATED OR EDITED
F	= FIRST OBJECT TO BE REVIEWED WHEN CYCLING
L	= LAST OBJECT TO BE REVIEWED WHEN CYCLING
F$	= SCENARIO NAME
T$(I)	= TIME MULTIPLIER AND STARTING TIMES

DEDIT PROGRAM

4.1 GENERAL INFORMATION

The DEDIT program generates and edits the decision alternatives or DSP (Decision String Phrases) that a participant may choose.

4.2 USING DEDIT

4.2.1 RUN DEDIT from the volume specified in VEDIT.

4.2.2 Command Menu lists file name, C for catalog, and Q for quit. By typing in a file name (which accesses a decision string phrase file), an existing file may be changed. Most Yugoslav Dilemma file names use the format Dr (where r is a one to four digit number). Some file names use the format Dr.@; this is used to create new decisions strings.

4.3 CREATING FILES

Figures 2 and 3 are graphic layouts of the DSP Files. Figure 2 illustrates the general layout and Figure 3 provides a specific example.

4.3.1 CLASS FILES

The top level is the CLASS File and it is given the name "D." It is for broad classifications such as Economic, Political, or Military. The phrases are not used to make sentences. When the CLASS File is created, it creates a VERB File for each phrase in the CLASS File. The phrase must end in a question mark (?) for information search classes.

```
EXAMPLE:  D:1  ECONOMIC
          D:2  POLITICAL
          D:3  MILITARY
          D:4  COVERT OPERATIONS
          D:5  PUBLIC OPINION
          D:6  INFORMATION SEARCH?
          creates 6 VERB Files:  D1.@, D2.@, D3.@, D4.@,
          05.@, and 06.@.
```

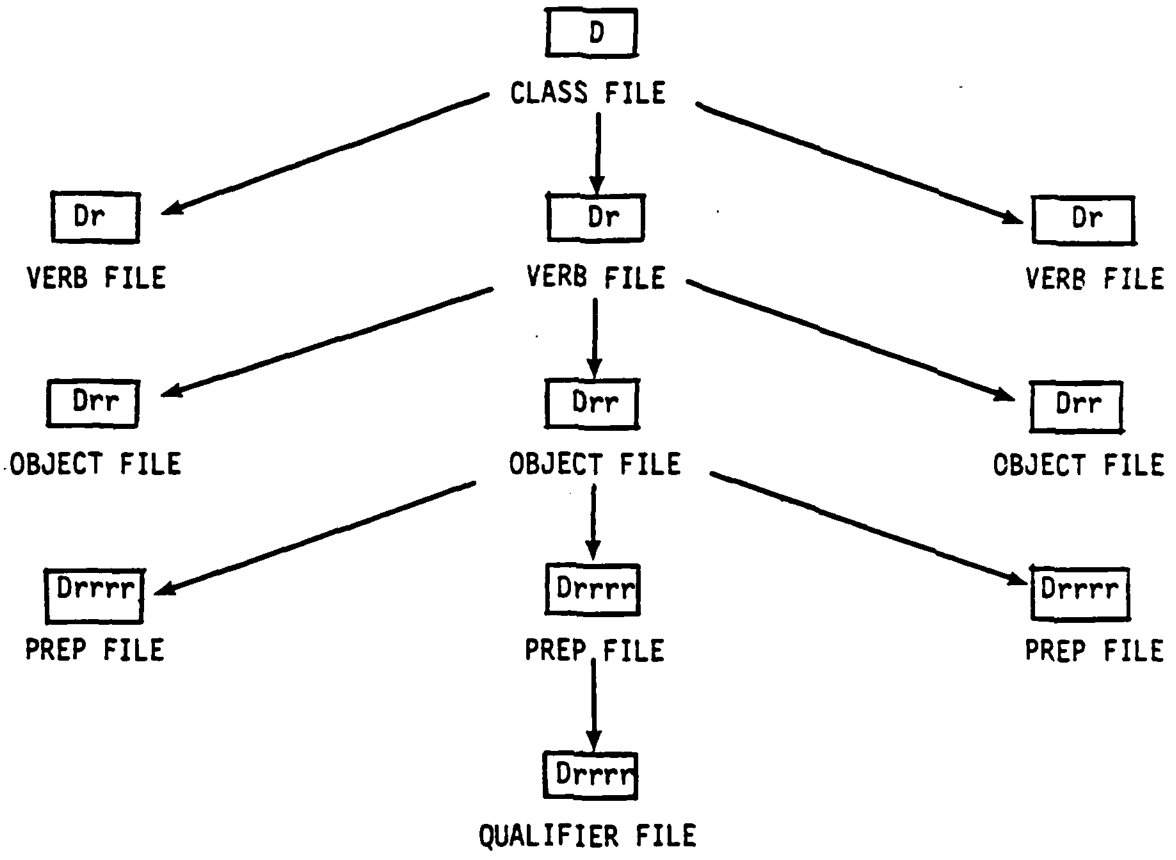


Figure 2. General titles of decision string phrase files

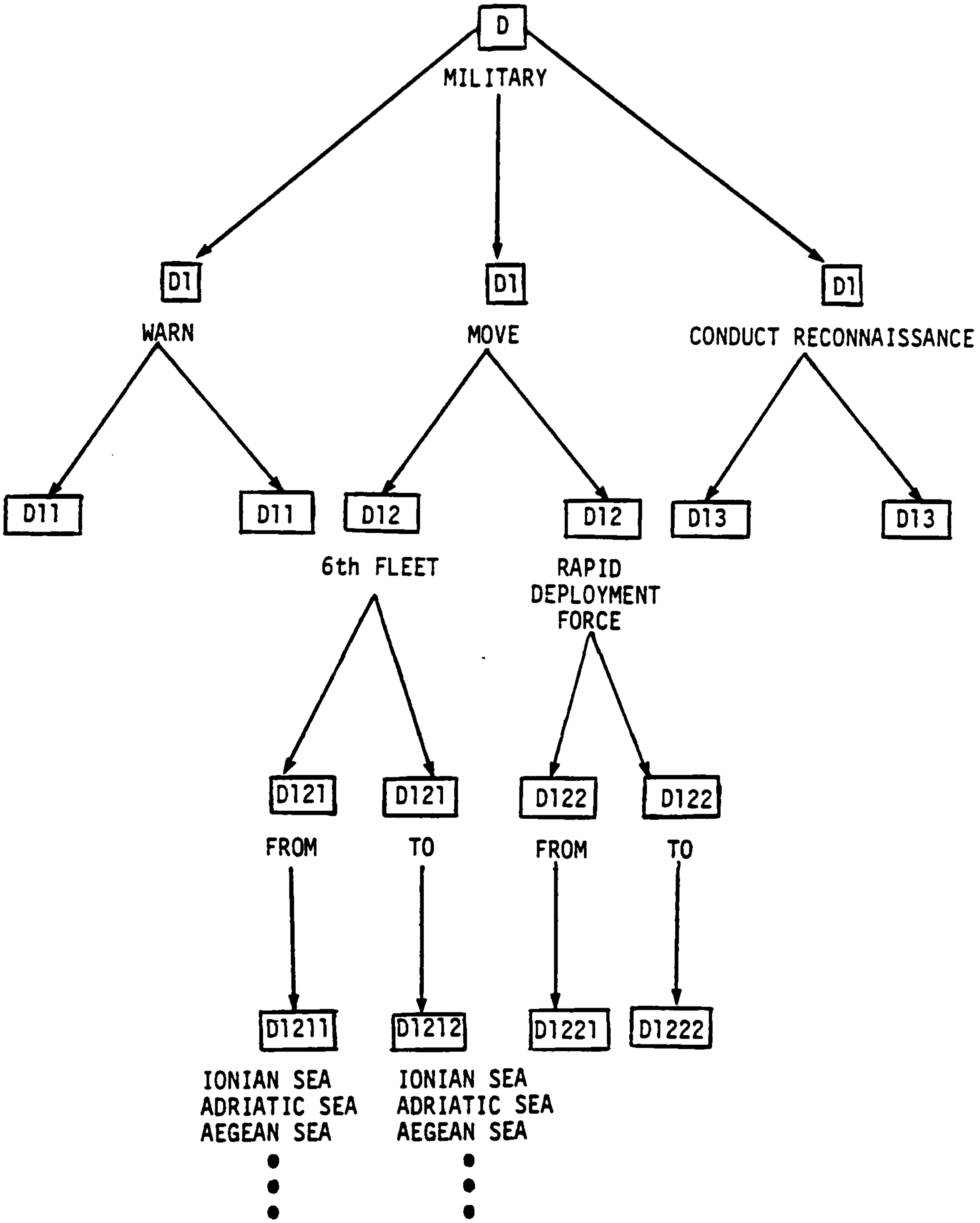


Figure 3. Example of decision string phrase file from the Yugoslav Dilemma

4.2.3 VERB FILES

The second level is composed of VERB Files with names of "Dr" where r is 1 to 9, depending on the number of phrases in the CLASS File. The VERB Files define the actions to be taken and are used as the beginning of the sentence built by SIM. When the VERB File is created it creates an OBJECT File for each VERB phrase. A VERB TYPE can be attached to the verb to define the action to be taken by SIM. The VERB TYPE is added to the end of the verb after a semi-colon (;). VERB TYPE 1 is a move verb. If no verb type is found or it is zero, no action will be taken.

```
EXAMPLE:  D1:1 WARN
          D1:2 MOVE; 1
          creates 2 OBJECT FILES, D11.@, and D12.@.
```

4.3.3 OBJECT FILES

The third level files are the OBJECT Files. They are given names of "Drr" where rr is 11 to 99, depending on the number of phrases in the VERB and CLASS files. The OBJECT phrases are used as the second part of the sentence created from the decision.

```
EXAMPLE:  D12:1 6th Fleet
          D12:2 Rapid Deployment Force
          creates a number of PREPosition
          and QUALIFIER Files.
```

4.3.4 PREP FILES

The PREP Files are tied to the QUALIFIER Files and have names of Drrr, where rrr is 111 to 999. They may contain a word, however, the file may be left blank. The prepositions are used as deletable parts of decision sentence building.

```
EXAMPLE:  D121:1  FROM
          D121:2  TO
          creates no new Files.
```

4.3.5 QUALIFIER FILES

QUALIFIER Files are the "who, what, and how" qualifiers for the OBJECT Files and have names of "Drrrr" where rrrr is 1111 to 9999. Up to nine Qualifiers may be attached to an OBJECT File and must be preceded by a PREP File. The Preposition for a Qualifier may be attached to the word or be put in the PREP File. The QUALIFIERS are added to the end of the sentences being built. At the end of QUALIFIERS, separated by symbols (#;&@) are ATTACHMENTS that are used by the SIM program to calculate movement and other actions. The ATTACHMENTS are defined in Section 4.3.6 and 4.3.7.

EXAMPLE:

D1211:1	Ionian Sea #1
D1211:2	Adriatic Sea #2
D1211:3	Aegean Sea #3

NOTE: The D1111 Files go with the D111:1 Files in 4.3.4; the D1112 Files go with the D111:2 Files; and D1113 with D111.3 Files.

4.3.6 ATTACHMENTS (#;&@)

ATTACHMENTS are information attached to QUALIFIER Files that allow calculation in SIM. No specific place has been set aside for the attachments. SIM uses them wherever found in a decision string.

# SYMBOL defines an object by number. The number ties the object to the definition in LEDIT.

;a;b SYMBOLS are used to define the amount of time in minutes that a decision will require. The calculation is made with (a+bx) where a is a fixed number of minutes, b is a rate of speed, and x is a computer-calculated (LEDIT) distance from origin to destination. If ;b does not appear, the calculation will be a fixed number of minutes. For example, if a decision string ending contains ;60, the response will be available (not necessarily delivered) in 60 minutes. If the string ends in ;120;4, th response will be available in 120 minutes + 4 (distance).

& SYMBOL defines the destination object number. The number ties the object to the definition in LEDIT. If the number following "&" is a zero, SIM will request a QUADRANT.

@ SYMBOLS define how the account files are to be manipulated. See 4.3.7.

4.3.7 ACCOUNT MANIPULATION ATTACHMENTS

ACCOUNT ATTACHMENTS are delimited by @ SYMBOLS on QUALIFIER Files. An account command is a number of characters surrounded by @ symbols. ACCOUNT ATTACHMENTS must be the LAST type of attachment appearing in a string. In order to use accounts they must be created using the AEDIT program to create an ATBL file. TEMPORARY accounts mentioned below are used to tie two different QUALIFIERS together in a SIM decision string. The ACCOUNT COMMAND consists of the following seven fields.

4.3.7.1 FIELD 1: A @ symbol.

4.3.7.2 FIELD 2: A ! symbol indicates an immediate action on the account. A ? symbol indicates action when the response message is shown on the screen.

4.3.7.3 FIELD 3: The Action indicator:
+ Add source account to destination account.
- Subtract source account from destination account.
x Multiply source account by destination account and store in destination account.
= Store source account into destination account. This works for either numbers or character accounts.

NOTE: IF A TEMPORARY ACCOUNT FROM A TO J APPEARS IN FIELD 4, FIELD 3 MUST NOT BE USED.

4.3.7.4 FIELD 4: The source account number from ATBL, a temporary account from A to J, or Q for a temporary QUADRANT account.

4.3.7.5 FIELD 5: A > symbol.

4.3.7.6 FIELD 6: The destination account number from ATBL, a temporary account from A to J, or Q for temporary QUADRANT account. A TEMPORARY account ties two QUALIFIER FILES together.

4.3.7.7 FIELD 7: A @ symbol. A destination account of 0 causes a value or string to be printed on the screen.

4.4 PUTTING DATA INTO A NEW FILE (Dr.@ Command)

4.4.1 DEDIT will ask how many phrases are to be entered. A number from 1 to 9 is acceptable.

4.4.2 Each phrase is then entered and terminated by a RETURN.

4.4.3 DEDIT then writes a new file with the name "Dr" and deletes the empty "Dr.@" file on the disk.

4.4.4 DEDIT writes new files at the next lower level (if any) with the name "Drx.@" where x is a number from 1 to the number of phrases.

EXAMPLE: D12.@ command would produce the following files if there were four phrases: D12, D121.@, D122.@, D123.@, D124.@. It would delete the D12.@ file.

4.4.5 Note that Dr.@ files are empty files used to indicate that more files remain to be filled. When all of the files with @ extensions have disappeared, all of the decision strings have been completed.

4.5 EDITING EXISTING FILES (Dr Command)

4.5.1 The RECORDS currently in the file will be displayed.

4.5.2 Enter the number of the record to be corrected. The cursor will move up to the beginning of the line to be edited. The new record can then be entered from the keyboard or be corrected using the cursor correction techniques specified in the APPLESOFT MANUAL using the left and right arrows and the ESC key with I, J, K, and M. Hit RETURN when the record is correct.

4.5.3 The number of phrases can be increased with an "I" Command or decreased with a "D" Command. Use of the "D" Command may leave some useless files on the disk but they will not cause a problem to the programs.

4.6 FILE

4.6.1 INPUT/OUTPUT FILES

NAME: Dr
where r is 1 to 9999 or blank.
SOURCE: DEDIT
EXAMPLE: D212

4.7 PROGRAM LISTING

DEDIT

COMMENTS

DEDIT PROGRAM START

```
1  HIMEM: 22479
2 LT = 22480: REM   1/25/83 2210
10  TEXT :REM DEDIT
90  ONERR  GOTO 9000
100  GOSUB 8000
110  PRINT : PRINT : GOTO 205
200  PRINT : HOME
```

ACCEPTS INITIAL USER INPUT
C TO CATALOG, Q TO QUIT
D@ TO ENTER NEW TEXT PHRASE
Dxxx TO RETRIEVE EXISTING
TEXT PHRASES

```
205  INPUT "ENTER FILE NAME OR Q OR
       C:";F$: IF F$ = "" THEN 205
206  IF F$ = "Q" THEN  END
207  IF F$ = "C" THEN  PRINT
       D4$"CATALOG": GOTO 205
208 NF =  VAL ( MID$ (F$,2,1)) + 7
210  IF F$ <  > "D.@" THEN 220
212  PRINT D4$"BSAVE"F$",A"LT",L1"VS$(7)
220  IF  RIGHT$ (F$,1) <  > "@" THEN 300
225  ONERR  GOTO 590
230  PRINT D4$"VERIFY"F$VS$(NF)
240  GOSUB 7000
250  GOTO 200
```

ALLOWS USER TO ADD, DELETE,
OR MODIFY EXISTING TEXT
PHRASES.

```
300  ONERR  GOTO 590
301  PRINT D4$"BLOAD"F$",A"LT;VS$(NF):LL
       =  PEEK (43616) + 256 *  PEEK
       (43617)
303 BL = 1:L = 0: GOTO 308
305  ONERR  GOTO 600
306  PRINT D4$"VERIFY"F$;VS$(NF): PRINT
       D4$"READ"F$
307 BL = 0:L = 1
308  GOSUB 9100
309  HOME : PRINT "FILE="F$: PRINT
310  FOR I = 0 TO A: PRINT ":"I +
       1"=";:V(I) =  PEEK (37)
320  PRINT D$(I): NEXT I: PRINT
       D4$"CLOSE"
330  PRINT : PRINT "ENTER RECORD # TO
       CORRECT, I TO INCREASE,D TO
       DECREASE, RETURN TO QUIT": GET A$
332  PRINT A$
334  IF A$ =  CHR$ (13) THEN 400
338  IF A$ <  > "D" THEN 344
340  PRINT "WARNING-YOU ARE
       DECREASING THE SIZE OF  THE
       MATRIX, OK? (Y/N):";: GET B$:
       PRINT B$: IF B$ <  > "Y" THEN 309
342 L = 1:A = A - 1: GOTO 309
344  IF A$ = "I" THEN 354
350 J =  VAL (A$) - 1: IF J < 0 OR (J >
       A) THEN  PRINT "": GOTO 330
352  IF (J <  = A) THEN L = 1: GOTO 360
353  GOTO 330
354  PRINT "WARNING-YOU ARE
       INCREASING THE SIZE OF  THE
       MATRIX, OK? (Y/N):";: GET B$:
       PRINT B$: IF B$ <  > "Y" THEN 309
```

```
355 A = A + 1:L = 1
356  INPUT "ENTER:";D$(A): IF D$(A) = ""
       THEN 356
357  IF  VAL ( MID$ (F$,2)) )  = 1000
       THEN 309
358 T$ = F$ +  STR$ (A + 1) + ".@":NQ =
       VAL ( MID$ (T$,2,1)) + 7: PRINT
       D4$"BSAVE"T$",A"LT",L1"VS$(NQ)
359  GOTO 309
360  VTAB V(J) + 1: HTAB 3: INPUT D$(J):
       GOTO 309
400  IF  NOT L THEN 500
402  ONERR  GOTO 410
405  REM :?  D4$"DELETE"F$
410  POKE LT,A:LM = LT
415  ONERR  GOTO 600
420  FOR I = 0 TO A:LM = LM + 1: POKE LM,93
421 LD =  LEN (D$(I)): FOR K = 1 TO
       LD:LM = LM + 1: POKE LM, ASC (
       MID$ (D$(I),K,1)): NEXT K
430  NEXT I:LM = LM + 1: POKE LM,13
440  PRINT D4$"BSAVE"F$",A"LT",L"LM - LT
       + 1;VS$(NF)
500  GOTO 200
590  PRINT : PRINT : PRINT "CAN'T FIND
       "F$" ON ("VS$(NF)")"
600  PRINT "ERROR ON DISC READ.
       HIT ANY KEY TO GO:": GET A$: GOTO
       200
6999  END
-----------------------------------------------------------------------
                                                    ACCEPTS INPUT OF NEW TEXT
                                                    PHRASES.
7000  PRINT "HOW MANY PHRASES";: GET N:
       PRINT N
7010 N = N - 1: FOR I = 0 TO N: PRINT
       "INPUT PHRASE #"I + 1;: INPUT
       D$(I): NEXT I
7015 LL = LT: POKE LL,N
7020  HOME :L = 0: FOR I = 0 TO N: PRINT
       I + 1"="D$(I): IF  LEN (D$(I)) >
       L THEN L =  LEN (D$(I))
7025  IF  NOT ( LEN (D$(I))) THEN  PRINT
       "ERROR, NO PHRASE": GOTO
       7000
7026  NEXT I
7030  PRINT : PRINT "OK? (Y/N):";: GET
       A$: PRINT A$
7040  IF A$ = "N" THEN 7000
7050  IF A$ <  > "Y" THEN 7030
7055 T$ =  LEFT$ (F$, LEN (F$) - 2)
7056 NF =  VAL ( MID$ (F$,2,1)) + 7
7060  POKE LL,N: FOR I = 0 TO N:LL = LL
       + 1: POKE LL,93
7061 LD =  LEN (D$(I)): FOR B = 1 TO
       LD:LL = LL + 1: POKE LL, ASC (
       MID$ (D$(I),B,1)): NEXT B
7065  IF  VAL ( MID$ (T$,2)) > 1000 THEN
       7080
7066 B$ = T$ +  STR$ (I + 1) + ".@"
7067 NQ =  VAL ( MID$ (B$,2,1)) + 7
7070  PRINT
       D4$"BSAVE"B$",A"LT",L1"VS$(NQ)
7080  NEXT I:LL = LL + 1: POKE LL,13
7090  PRINT D4$"DELETE"F$VS$(NF): PRINT
       D4$"BSAVE"T$",A"LT",L"LL - LT +
       1;VS$(NF)
7099  RETURN
-----------------------------------------------------------------------
```

```
                             SETUP SUBROUTINE
---------------------------------------------------------------------
                                            DIMENSION VARIABLES
8000  DIM VS$(16)
8010 D4$ =  CHR$ (4)
8020  HOME : VTAB 8: HTAB 10: PRINT
       "DEDIT PROGRAM": PRINT : PRINT
---------------------------------------------------------------------
                                            READ V/SCENARIO FILE
8040  INPUT "ENTER SCENARIO:";X$
8041  PRINT D4$"VERIFYV/"X$"
8042  PRINT D4$"READV/"X$: FOR I = 0 TO
       16
8043  GET A$: IF  ASC (A$) = 13 THEN
       8045
8044 VS$(I) = VS$(I) + A$: GOTO 8043
8045  NEXT I: PRINT : PRINT D4$"CLOSE"
8046  PRINT " "
8100 LB$ =  CHR$ (91):CR$ =  CHR$ (13):
       FOR I = 0 TO 99: READ CM$(I)
8101  IF CM$(I) <  > "$" THEN  NEXT I
8102 NC = I - 1
8110  DATA  DELETE,REPLACE,PRINT
8111  DATA $
8199  GOTO 8999
---------------------------------------------------------------------
                                            DEBUG ROUTINE
8200  PRINT D4$"OPEN TEST"
8210  PRINT D4$"WRITE TEST"
8220 A$ = "40:A1]A2]A3]A4]A5]": PRINT A$
8230  PRINT D4$"CLOSE"
8240  PRINT D4$"OPEN TEST"
8250  PRINT D4$"READ TEST"
8262  GET A$: GET B$:A =  VAL (A$):B =
       VAL (B$): GET A$
8263  FOR J = 0 TO B: FOR I = 0 TO
       A:D$(I,J) = ""
8264  GET A$: IF A$ <  > "]" THEN
       D$(I,J) = D$(I,J) + A$: GOTO 8264
8265  PRINT D$(I,J): NEXT I: NEXT J
8270  PRINT D4$"CLOSE"
8280  END
8999  RETURN
---------------------------------------------------------------------
                                            ERROR ROUTINE
9000 ER =  PEEK (222): PRINT
       "ERROR="ER
9010  IF ER = 6 THEN  PRINT "FILE
       NOT FOUND": GOTO 205
9030  END
---------------------------------------------------------------------
                                            SPLITS STRING INTO SMALLER
                                            STRINGS
9100  IF BL THEN 9105
9102  GET A$:A =  VAL (A$): GET B$
9103  GOTO 9110
9105 A =  PEEK (LT):LM = LT + 1
9110  FOR I = 0 TO A:D$(I) = ""
9112  IF BL THEN LM = LM + 1:A$ =  CHR$
       ( PEEK (LM)): GOTO 9120
9113  GET A$
9120  IF A$ <  > CR$ AND A$ <  > "]"
       THEN D$(I) = D$(I) + A$: GOTO
       9112
9130  NEXT I: PRINT
9140  RETURN
---------------------------------------------------------------------
                                            UTILITY PROGRAMS NOT
                                            EXECUTED BY DEDIT
```

```
50000 I$ =  CHR$ (9):Q$ =  CHR$ (27) D$
       =  CHR$ (4):S$ =  CHR$ (31):M$ =
       CHR$ (30):L$ =  CHR$ (29):NC$ =
       CHR$ (2):EX$ =  CHR$ (1)
50002  PRINT D$"PR#0"
50005  PRINT D$"PR#1"
50006  PRINT Q$"J,0,960,$"
50007  PRINT Q$"B,6,$"
50010  PRINT I$0"N"
50020  PRINT Q$"R.2,$"
50030  PRINT M$NC$
50100  END
55000 D$ =  CHR$ (4): PRINT D$"OPEN
       ADDLIST": PRINT D$"WRITE
       ADDLIST": LIST : PRINT D$"CLOSE".
       END
```

KEY VARIABLES

X$ = SCENARIO NAME
F$ = INITIAL USER INPUT (DECISION ALTERNATIVE TO BE EDITED)

AEDIT PROGRAM

5.1 GENERAL INFORMATION

The AEDIT program produces the account records for responsive messages that are used by the SIM program.

5.2 USING AEDIT

5.2.1 RUN AEDIT from the volume defined by VEDIT.

5.2.2 Enter the scenario.

5.2.3 Command Menu:

E=Edit a record.
C=Cycle list of records.
P=Printer ON.
N=No Printer.
Q=Quit.
?=Any other key gets above list.

5.3 PRODUCING AN ACCOUNT RECORD

5.3.1 The records are produced by using the EDIT command.

WARNING: TRY TO KEEP THE RECORD NUMBERS AS LOW AS POSSIBLE, OTHERWISE MANY EXCESS RECORDS COULD BE CREATED. Record numbers can be any number greater than 1. Record 1 contains the number of records already created.

5.3.2 Each account record can have up to 200 characters (five lines) in it. An account can have either a number in it or any character message (including non-changeable numbers).

5.4 LISTING RECORDS

The records can be listed using the "C" command. AEDIT will show the current first and last record numbers. Select the range of records to be displayed.

5.5 FILES

The AEDIT program creates (if necessary) and uses a random access disk file of 200 characters in each record.

5.5.1 INPUT/OUTPUT FILE

NAME: ATBL/s
where s is scenario name
SOURCE: AEDIT
EXAMPLE: ATBL/YUGOSLAV DILEMMA

5.6 ENDING PROGRAM

Select the Q (QUIT) option. All new records are immediately written to disk

5.7 PROGRAM LISTING

```
                               AEDIT
                                                  COMMENTS
----------------------------------------------------------------------
                                             AEDIT PROGRAM START
10  REM  AEDIT
11  HOME
12 SZ = 200
50 D4$ =  CHR$ (4)
60  GOTO 1800
100  GOSUB 8000
----------------------------------------------------------------------
                                             CHECKS AND ACCEPTS USER
                                             COMMANDS
110  HOME : PRINT "MENU -- LARGEST="NN
200  PRINT "COMMAND (";: FOR I = 0 TO
       NA: PRINT  LEFT$ (C$(I),1);: NEXT
       I: PRINT "):";
210  GET A$: PRINT A$
212  FOR C = 0 TO NA: IF A$ =  LEFT$
       (C$(C),1) THEN 250
215  NEXT C
220  PRINT : PRINT : PRINT "COMMANDS:":
       FOR I = 0 TO NA: PRINT C$(I):
       NEXT I: GOTO 200
250  ON C + 1 GOTO
       300,600,1200,1500,3000
----------------------------------------------------------------------
                                             CODE FOR EDIT COMMAND
300  PRINT : PRINT "ENTER #:";: INPUT T
310  IF T < 2 OR T > 499 THEN  PRINT
       "ERROR, MUST BE 2 TO 499":
       GOTO 200
311  IF T > NN THEN 500
320  GOSUB 10000
401  PRINT "CURRENT MESSAGE STRING=":
       PRINT M$: PRINT
402  PRINT "DO YOU WANT TO CHANGE IT?
       (Y/N):";: GET A$: PRINT A$: PRINT

403  IF A$ = "N" THEN 599
404  IF A$ <  > "Y" THEN 402
410  PRINT "ENTER MESSAGE STRING UP TO
       "SZ" CHARS.:": INPUT M$
411  IF T > NN THEN NN = T: GOSUB 11000
420  IF M$ = "" THEN 410
430 M$ =  LEFT$ (M$,SZ)
440  GOSUB 12000
450  GOTO 320
500  PRINT "DO YOU WANT TO ADD A NEW
       RECORD? (Y/N):";: GET A$: PRINT
       A$: PRINT
503  IF A$ = "N" THEN 200
504  IF A$ <  > "Y" THEN 500
509 F = NN + 1:NN = T:L = NN
510  PRINT D4$"OPEN"N$F$",L"SZ
520  PRINT D4$"WRITE"N$F$",R1"
530  PRINT NN
540  GOSUB 2000
550  GOTO 320
599  PRINT D4$"CLOSE": GOTO 200
----------------------------------------------------------------------
                                             CODE FOR CYCLE COMMAND
600  PRINT "CYCLE"
625  PRINT "LAST RECORD="NN: PRINT
626  PRINT "HIT ! TO STOP LISTING, ANY
       OTHER TO     STOP AND RESTART
       SCROLLING"
627  PRINT
630  INPUT "INPUT FIRST, LAST FOR
```

```
     CYCLE:";F,L
632  IF F > L OR (F < 1 OR L > 500) THEN
     PRINT "ERR": GOTO 630
639  POKE  - 16368,0
640  FOR T = F TO L: PRINT T;
641  INVERSE : PRINT
     "----------------------------".
     NORMAL
642  PRINT D4$"OPEN"N$F$",L"SZ
645  PRINT D4$"READ"N$F$",R"T: INPUT M$
646  PRINT D4$"CLOSE"
650  PRINT M$
660  PRINT
661  GOTO 699
690 X =  PEEK ( - 16384): IF X < 128
     THEN 699
691  POKE  - 16368,0
692  IF X = 161 THEN 200
693 X =  PEEK ( - 16384): IF X < 128
     THEN 693
694  POKE  - 16368,0
699  NEXT T
700  GOTO 200
------------------------------------------------------
                                   PRINTER ON/OFF COMMANDS
1200  PRINT D4$"PR#1": GOTO 200
1500  PRINT D4$"PR#0": GOTO 200
------------------------------------------------------
                                   IDENTIFIES AND READS THE
                                   SCENARIO OR PARTICIPANT FILE
                                   THAT IS TO BE CREATED OR
                                   EDITED.
1800  PRINT D4$"CLOSE"
1801 VV$ = "":N$ = "ATBL/":TY$ =
     "SCENARIO"
1802  PRINT "SCENARIO OR PARTICIPANT?
      (S/P):";: GET A$: PRINT A$: PRINT
      : IF A$ <  > "P" AND A$ <  > "S"
      THEN 1802
1803  IF A$ <  > "P" THEN 1809
1804 N$ = "A/":TY$ = "PARTICIPANT"
1805  INPUT "ENTER VOLUME NUMBER FOR A/P
      FILE:";N
1806 W$ = ".V" +  STR$ (N)
1809  ONERR  GOTO 1850
1810  PRINT "ENTER "TY$" NAME:";: INPUT
      F$
1820  IF F$ = "" THEN 200
1830  PRINT D4$"OPEN"N$F$",L"SZ
1835  PRINT D4$"READ"N$F$",R1"
1840  INPUT NN
1841  PRINT "HIGHEST RECORD="NN
1849  GOTO 1890
1850  INPUT "NEW SCENARIO? (Y/N) ";AN$
1851  IF AN$ = "N" THEN 1800
1852  IF AN$ <  > "Y" THEN 1800
1855  POKE 216,0
1860 NN = 1
1865  IF PN = NN THEN 1890
1866  PRINT D4$"OPEN"N$F$",L40"
1870  PRINT D4$"WRITE"N$F$",R1"
1880  PRINT NN
1881 F = 2:L = 2
1882  GOSUB 2000
1890  PRINT D4$"CLOSE"N$F$"
1899  POKE 216,0: GOTO 100
1910  RETURN
------------------------------------------------------
```

```
                                                 WRITES A NEW SCENARIO TO DISK
1999  PRINT D4$"OPEN"N$F$",L40"
2000  FOR I = F TO L
2010  PRINT D4$"WRITE"N$F$",R"I
2020  PRINT  ????? EMPTY #"I" ?????"
2030  NEXT I
2050  RETURN
3000  PRINT D4$"PR#0": END
----------------------------------------------------------------------
                              SETUP SUBROUTINE
----------------------------------------------------------------------
                                                 DIMENSION VARIABLES
8000  DIM T%(200),T$(200)
----------------------------------------------------------------------
                                                 INITIALIZE VARIABLES
8010  FOR NA = 0 TO 999: READ C$(NA)
8020  IF C$(NA) = "$" THEN 8050
8025  NEXT NA
8030  DATA EDIT,CYCLE,PRINTER ON,NO
       PRINTER,QUIT
8040  DATA $
8050 NA = NA - 1
8999  RETURN
----------------------------------------------------------------------
                                                 READ A DISK FILE MESSAGE
10000 SZ = 200: PRINT
       D4$"OPEN"N$F$",L"SZ
10010  PRINT D4$"READ"N$F$",R"T
10015  INPUT M$
10020  PRINT D4$"CLOSE"F$
10030  RETURN
----------------------------------------------------------------------
                                                 WRITES THE CURRENT NUMBER
                                                 OF TOTAL RECORDS ON RECORD
                                                 NUMBER 1
11000 SZ = 200: PRINT
       D4$"OPEN"N$F$",L"SZ
11010  PRINT D4$"WRITE"N$F$",R1"
11020  PRINT NN
11030  PRINT D4$"CLOSE"F$
11040  RETURN
----------------------------------------------------------------------
                                                 WRITES A SPECIFIC MESSAGE
                                                 STRING
12000 SZ = 200: PRINT
       D4$"OPEN"N$F$",L"SZ
12010  PRINT D4$"WRITE"N$F$",R"T
12020  PRINT M$
12030  PRINT D4$"CLOSE"F$
12040  RETURN
----------------------------------------------------------------------
                                                 UTILITY PROGRAMS NOT
                                                 EXECUTED BY AEDIT
30000  INPUT B$: PRINT B$
50000 I$ =  CHR$ (9):Q$ =  CHR$ (27):D$
       =  CHR$ (4):S$ =  CHR$ (31):M$ =
       CHR$ (30):L$ =  CHR$ (29):NC$ =
       CHR$ (2):EX$ =  CHR$ (1)
50002  PRINT D$"PR#0"
50005  PRINT D$"PR#1"
50006  PRINT Q$"J,0,960,$"
50007  PRINT Q$"B,6,$"
50010  PRINT I$0"N"
50020  PRINT Q$"R,2,$"
50030  PRINT M$NC$
50100  END
55000 D$ =  CHR$ (4): PRINT D$"OPEN
       ADDLIST": PRINT D$"WRITE
       ADDLIST". LIST : PRINT D$"CLOSE":
```

KEY VARIABLES

```
NN    = TOTAL NUMBER OF RECORDS IN A FILE
SZ    = MAXIMUM NUMBER OF CHARACTERS ALLOWED IN A RECORD
T     = CURRENTLY ACCESSED RECORD NUMBER
M$    = CURRENT MESSAGE STRING
F     = FIRST RECORD TO BE REVIEWED WHEN CYCLING
L     = LAST RECORD TO BE REVIEWED WHEN CYCLING
N$F$  = PARTICIPANT OR SCENARIO FILE THAT IS CURRENTLY BEING READ
         OR WRITTEN
C$(I)= MENU COMMANDS
```

SIM PROGRAM

6.1 GENERAL INFORMATION

The SIM Program is the operational program in the simulation sequence. It allows the participant to make decisions in response to messages received.

6.2 PREREQUISITES

Prior to running SIM, the proper files must be generated from the programs described in Sections 2 through 5 of this document. See Section 6.4.1 for files required for input to SIM.

6.3 USING SIM

Detailed instructions for starting the simulation can be found in Criswell, Unger, Swezey and Streufert (1983). In general, the simulation can be started with the following commands:

```
BLOAD RUNTIME, VØ30
BLOAD SIMYD.OBJ (or SIMSTORM.OBJ if
  the practice session is to be run)
CALL 6064
```

or

```
EXEC YD,VØ3Ø
```

6.4 FILES

6.4.1 INPUT FILES

NAME: SIM
SOURCE: Program

NAME: LOC/s
where s is the scenario name
SOURCE: LEDIT
EXAMPLE: LOC/STORM

NAME: Dr
where r is a reference number from
1 to 9999 or blank
SOURCE: DEDIT
EXAMPLE: D121

NAME: ATBL/s
where s is the scenario
SOURCE: AEDIT
EXAMPLE: ATBL/STORM

NAME: TEXT.Mn
where n is a message number from
1 to 999
SOURCE: APPLE WRITER (TEDITOR)
EXAMPLE: TEXT.M25

NAME: TS#/s
where s is the scenario name
SOURCE: TEDIT
EXAMPLE: TS#/STORM

NAME: TM/s
where s is the scenario name
SOURCE: TEDIT
EXAMPLE: TM/STORM

6.4.2 INPUT/OUTPUT FILES

NAME: Rt/p
where t is the time, p is
participant code
SOURCE: SIM
EXAMPLE: R5/JOHN DOE
CONTENTS: Message

Real Time Minutes

For each cycle including
Message cycle:
Real Time of Day
Number of Decision Cycles

Real Time of Day   Previous Decisions

Real Time of Day   Previous Messages

SIM

```
                                                          COMMENTS
---------------------------------------------------------------------------------------
                                                          START PROGRAM
1  HIMEM: 37000:I = 200:J = I:K = J:QI =
       I:QJ = I:SZ = I:QS$ = "*":IS =
       128: REM   CHANGE
       STATEMENT-VARIABLE OM% TO EQUAL
       IS
2 LT = 37001: TEXT :CV = 2: REM
       12/28/83
10  HOME : VTAB 10: HTAB 10: PRINT
       "PROGRAM SIM"
11  GOSUB 8000
20  GOTO 500
---------------------------------------------------------------------------------------
                                                          COMPUTES AND DISPLAYS
                                                          SIMULATION TIME
30  PRINT : PRINT D4$"IN#4": PRINT
       D4$"PR#4": INPUT " ";U$: PRINT
       D4$"PR#0": PRINT D4$"IN#0":U$ =
       LEFT$ (U$,14): RETURN
40  POKE 34,0: POKE 35,1:T$ = "TIME= ":
       IF DT(2) < 10 THEN T$ = T$ + "0"
41  VTAB 1: HTAB 1
42 T$ = T$ +  STR$ ( INT (DT(2))): IF
       DT(1) < 10 THEN T$ = T$ + "0"
44 T$ = T$ +  STR$ ( INT (DT(1))) + ":":
       IF DT(0) < 10 THEN T$ = T$ + "0"
46 T$ = T$ +  STR$ ( INT (DT(0))) + "
       " +  STR$ ( INT (DT(3))) + " " +
       MO$(DT(4)) + " " +  STR$ (DT(5))
       + "   "
48  VTAB 1: HTAB 1: PRINT T$;: POKE
       34,0: POKE 35,1: HTAB 1: VTAB 2:
       PRINT DH$;: POKE 34,CV: POKE
       35,23: RETURN
50 QD(QN) = DT(QN) + T
58  FOR QB = QN TO 4:QI =  INT ((QD(QB)
       - (QB > 1)) / QS(QB))
59 QD(QB) = QD(QB) - QI * QS(QB):QD(QB +
       1) = DT(QB + 1) + QI: NEXT QB
61  IF QD(2) > 23 THEN QD(2) = 0:QD(3) =
       QD(3) + 1
62  IF QD(3) > 28 THEN QD(3) = 1:QD(4) =
       QD(4) + 1
63  IF QD(4) > 12 THEN QD(4) = 1:QD(5) =
       QD(5) + 1
70  RETURN
---------------------------------------------------------------------------------------
                                                          START OF SIMULATION
500  GOSUB 40: HOME : VTAB 20: PRINT
       "HIT ! TO START SIMULATION":PV =
       0:PR = 0
510  GET A$: IF A$ <  > "!" THEN 510
511  GOSUB 7400
513 CV = 2:VT = 99999: POKE 34,CV: HOME
514  POKE SW%,1: PRINT : PRINT PR$:
       PRINT X$CD$: PRINT PO$
519  PRINT : HOME
520  GOSUB 30:LT$ = U$
---------------------------------------------------------------------------------------
                                                          COMPUTES RELATIVE TIMES
                                                          TO INITIALIZE CLOCK
540 DO%(0) =  VAL ( MID$
       (LT$,13,2)):DO%(1) =  VAL ( MID$
       (LT$,10,2)):DO%(2) =  VAL ( MID$
       (LT$,7,2))
560 A$ = U$: GOTO 640
```

```
600  GOSUB 30:A$ = U$
640  IF LT$ = A$ THEN 1160
660 LT$ = A$
680 DN%(0) =  VAL ( MID$
     (A$,13,2)):DN%(1) =  VAL ( MID$
     (A$,10,2)):DN%(2) =  VAL ( MID$
     (A$,7,2))
700 T = ((DN%(0) - DO%(0)) + (DN%(1) -
     DO%(1)) * 60 + (DN%(2) - DO%(2))
     * 3600)
710  IF T < 0 THEN T = T + 86400
720  IF SD% THEN T = SD:SD% = 0
740 RM = RM + T / 60: VTAB 1: HTAB 33:
     IF DM% THEN  PRINT  INT (RM);
741  VTAB 3: HTAB 1
750  IF (RM - VT) > .6 THEN CV = 2:VT =
     99999: POKE 34,CV: HOME
760 T = T * TM
820 QN = 0: GOSUB 50: FOR I = 0 TO
     5:DT(I) = QD(I): NEXT I
840  GOSUB 40: FOR I = 0 TO 2:DO%(I) =
     DN%(I): NEXT I
860 PK% =  ABS (T%(PM))
880  IF VT < 90000 OR PM >  INT (RM)
     THEN 1160
900  HOME : IF PK% < 10 THEN 980
-----------------------------------------------------------------------
                                                  PRINTS OUT MESSAGES AT THE
                                                  CORRECT TIME
910  IF PK% = 16 THEN QS$ = SM$:SM$ =
     "": GOTO 921
92C  PRINT : HOME : PRINT
     D4$"OPENR#";CD$VS$(4)",L500":
     PRINT D4$"READR#"CD$",R"PM: INPUT
     QS$: PRINT D4$"CLOSE"
921  PRINT PR$
925  GOSUB 9660: GOSUB 9620: GOSUB 9600:
     GOSUB 9690: GOSUB 9990
930 VT = RM
932  PRINT : PRINT
940  HOME
960  GOSUB 9630: GOTO 600
980  PRINT : HOME : PRINT
     D4$"OPENTS#/"X$",L40";VS$(3):
     PRINT D4$"READTS#/"X$",R"PM:
     INPUT QS$: PRINT D4$"CLOSE":
     GOSUB 9100
1000 J = PK% < 4
1005  IF PK% = 8 THEN PV = 1
1006  IF PK% = 9 THEN PV = 0
1020  IF QN = 0 AND J = 1 THEN 930
1040 K = 0: FOR L = J TO QN:QV(L) =
     QV(L):K = K +  NOT U%(QV(L)):
     NEXT L
1060  IF  NOT K THEN L = J: GOTO 1120
1080 K =  INT ( RND (1) * K): FOR L = J
     TO QN:K = K -  NOT U%(QV(L)): IF
     K < 0 THEN 1120
1100  NEXT L
1120  PRINT PR$
1140 QI = QV(L):T%(PM) = (1000 + QI) *
     SGN (T%(PM))
1145 U%(QV(L)) = 1:QS$ = ""
1148  GOSUB 9504: GOSUB 9660: GOSUB
     9620: GOSUB 9600: GOSUB 9690:
     GOSUB 9990
1150  IF PK% = 7 THEN  GOSUB 9610:CP =
     CP + 1: GOSUB 9630: GOTO 500
1155  GOTO 930
```

```
1160  VTAB CV + 2: HTAB 1: IF PV THEN
       POKE SB%,0: GOTO 600
1180  PRINT MM$
1200  IF  PEEK (B0%) > 127 THEN  GOSUB
       7100: POKE SB%,0: POKE SW%,1
1210 K =  PEEK (KB%): IF K < 128 THEN
       600
1220  POKE SB%,0
1240 CY = 0:B$ = "":QU% = 0
1260  IF K = 196 THEN 1280
1270  IF  PEEK (B0%) > 127 THEN  GOSUB
       7100
1272  GOTO 600
-----------------------------------------------------------------------
                                                 STARTS DECISION STRINGS
1280 F$ = "D": HOME :Y$ = "":NO =  -
       1:NW =  - 1:B$(CY) = "":UQ% =
       0:BB = CY
1300 L = 0
1310 SS$ = "":QA =  FRE (0)
1320 I =  VAL ( MID$ (F$,2,1)) + 7
1325  PRINT D4$"BLOAD"F$;VS$(I)",A"LT
1330 CV = 2:VT = 99999: POKE 34,CV
1340 R$(CY) = ""
1360 J = 0:QI = LT:NP(L) =  PEEK (QI):QI
       = QI + 1
1380 P$(0,L) = "": FOR I = 0 TO 9999:QI
       = QI + 1:QN =  PEEK (QI): IF QN =
       13 THEN 1460
1400  IF QN = 93 THEN J = J + 1:P$(J,L)
       = "": GOTO 1440
1420 P$(J,L) = P$(J,L) +  CHR$ (QN)
1440  NEXT I
1460  PRINT
1500  IF L = 3 THEN 1700
1520  IF L = 1 THEN  FOR M = 0 TO
       NP(L):VB(M) = 0:QS$ = P$(M,L):
       GOSUB 9100:VB(M) = QV(1):P$(M,L)
       =  MID$ (QS$,1,QE): NEXT M
1540 QS$ = AA$(CY > 0): FOR M = 0 TO
       NP(L):QS$ = QS$ + " " +  STR$ (M
       + 1) + "." + P$(M,L): IF M <
       NP(L) THEN QS$ = QS$ + ", "
1560  NEXT M: GOSUB 9800: FOR QI = 0 TO
       QN: PRINT QS$(QI): NEXT QI
1580  PRINT
1600  GOSUB 7500:A$ = S$
1620 V =  VAL (A$): IF V < 1 OR V > M
       THEN 1600
1630  IF L OR  RIGHT$ (P$(V - 1,0),1) <
       > "?" THEN 1640
1632 UQ% = 1: IF  NOT CY THEN QU% = 1
1640 NP(L) = V:F$ = F$ + A$
1660  IF (BB >  - 1) AND ((L > 1) OR ((L
       = 1) AND (( NOT UQ% AND CY) OR (
       NOT QU% AND  NOT CY)))) THEN Y$ =
       Y$ + P$(V - 1,L) + " "
1680 L = L + 1: IF L < 4 THEN 1320
1700  FOR K = 0 TO 8:EQ(0,K) = 0:EQ(1,K)
       = 0: NEXT K
1720 L = 0
1740 I =  VAL ( MID$ (F$,2,1)) + 7:
       PRINT : PRINT D4$"BLOAD"F$;L +
       1,VS$(I)",A"LT:R$(CY) = R$(CY) +
       ";" + F$ +  STR$ (L + 1)
1760 J = 0:QI = LT:PP(L) =  PEEK (QI):QI
       = QI + 1
1770 PP$(0,L) = "": FOR I = 0 TO 9999:QI
       = QI + 1:QN =  PEEK (QI): IF QN =
       13 THEN 1880
1780   IF QN = 93 THEN J = J + 1:PP$(J,L)
       = "": GOTO 1800
1790  PP$(J,L) = PP$(J,L) +  CHR$ (QN)
1800   NEXT I
1880   PRINT :QD = 0
1900  V =  VAL (P$(L,3)):C$ =  MID$
       (P$(L,3),(V > 0) + 1,99)
1920  P$ = AA$(CY > 0): FOR M = 0 TO
       PP(L):P$ = P$ + " " +  STR$ (M +
       1) + "." + C$: IF C$ <  > "" THEN
       P$ = P$ + " "
1930  QO = 0:QW = 0:QN = 0:WS$(M) = ""
1940  QS$ = PP$(M.L): GOSUB 9100
1960   IF QO THEN OB(M) = QO
1980   IF QW THEN WH$(M) = QW$
1990   IF (QA) THEN WS$(M) = QA$
2000   IF QN = 0 THEN 2060
2020  EQ(1.M) = 0:QD = 1
2040  EQ(0,M) = QV(1): IF QN > 1 THEN
       EQ(1.M) = QV(2)
2060  PP$(M.L) =  MID$ (QS$,1.QE)
2080  P$ = P$ + PP$(M,L)
2100   IF M < PP(L) THEN P$ = P$ + ","
2120   NEXT M
2140  QS$ = P$: GOSUB 9800: IF BB >  = 0
       THEN  FOR QI = 0 TO QN: PRINT
       QS$(QI): NEXT QI
----------------------------------------------------------------------
                                                 COMPUTES TIME REQUIRED TO
                                                 IMPLEMENT DECISION
2160   PRINT
2200   GOSUB 7500:QS$ = S$
2220  QV(0) =  VAL (QS$): IF QV(0) < 1 OR
       QV(0) > M THEN 2200
2240  QV(0) = QV(0) - 1: IF QO THEN OB(0)
       = OB(QV(0))
2260   IF QD THEN EQ(0,0) =
       EQ(0,QV(0)):EQ(1.0) = EQ(1.QV(0))
2270  SS$ = SS$ + WS$(QV(0))
2280   IF  NOT QW THEN 2300
2282  A$ = WH$(QV(0)):QW =  VAL (A$): IF
       NOT QW THEN 2290
2284  X = OM%(1.QW):Y = OM%(2,QW)
2285  B$(CY) = "": GOTO 2296
2290  R$(CY) = R$(CY) + "." + QS$ + "/":X
       =  - 1: IF BB < 0 THEN QS$ =
       "AA1": GOSUB 9200: GOTO 2293
2291   PRINT : PRINT "ENTER";: PRINT "
       QUADRANT:";: INPUT QS$: IF  LEN
       (QS$) > 2 THEN  GOSUB 9200
2292   IF X < 0 THEN  PRINT "ERROR, 2
       LETTERS + A NUMBER (I.E.AB12)":
       GOTO 2290
2293  B$(CY) = B$(CY) + " " + QS$:QQ$ =
       QS$
2294  X = X + OM%(1,0):Y = Y + OM%(2,0)
2296   REM
2300  QN = 0
2320   GOTO 2460
2460   IF BB >  - 1 THEN Y$ = Y$ + C$ + "
       " + PP$(QV(0),L) + B$(CY)
2470  R$(CY) = R$(CY) + "." + QS$
2480   IF BB >  - 1 THEN Y$ = Y$ + " "
2620  L = L + 1: IF L <  = NP(3) THEN
       1740
2640   IF CY THEN 3300
2650   PRINT :QS$ = Y$
2660   ON VB(NP(1) - 1) + 1 GOTO
```

```
       2720,2680,1960
2680   REM
2700   IF  NOT OM%(0,OB(0)) OR
       OM%(0,OB(0)) = 2 THEN  PRINT
       "ERR": GOTO 2750
2720 X = ((X - OM%(1,OB(0))) ^ 2) + ((Y
       - OM%(2,OB(0))) ^ 2): IF X THEN X
       =  SQR (X)
2740 T =  INT (EQ(0,0) + X * EQ(1,0) +
       .4999)
2750   PRINT : GOSUB 9800: IF  NOT QU%
       THEN  PRINT YD$
2760   FOR I = 0 TO QN: PRINT QS$(I):
       NEXT I
2780 OD =  INT (T / 1440.):OH =  INT ((T
       - OD * 1440.) / 60 ):I =  INT (T
       - OD * 1440. - OH * 60.)
2785   PRINT : PRINT "THIS DECISION CAN
       NOT BE IMPLEMENTED   BEFORE "OD"
       DAYS, "OH" HOURS & "I" MIN."
------------------------------------------------------------------------
                                                 ALLOWS DELAYING
                                                 IMPLEMENTATION OR
                                                 DELETEING DECISION
2790   PRINT
2800 QI = 906: GOSUB 9500: GET A$: PRINT
       A$
2820   IF A$ < > "A" THEN 2880
2840   PRINT : INPUT "ENTER ADDITIONAL
       DAYS,HOURS,MINUTES    SEPARATED
       BY COMMAS (EXAMPLE: 0,3,45)
       ?:";OD,OH,I:I = I + 1440 * OD +
       60 * OH: IF I < 0 THEN  PRINT
       "ERROR": GOTO 2800
2860 T = T + I: GOTO 2780
2880   IF A$ = "C" THEN 3610
2900   IF A$ < > "E" THEN 2800
2920 QN = 1: GOSUB 50
2960   GOSUB 9650
2980   IF  NOT QU% THEN B$(CY) = "YOUR
       DECISION TO " + QS$: GOTO 3000
2990   PRINT "YOU HAVE DECIDED TO
       REQUEST": PRINT "INFORMATION
       TO:":B$(CY) = "IN RESPONSE TO
       YOUR REQUEST TO " + QS$
3000   PRINT :QS$ = QS$ + "BY
       APPROXIMATELY ": IF QD(2) < 10
       THEN QS$ = QS$ + "0"
3020 QS$ = QS$ +  STR$ (QD(2)): IF QD(1)
       < 10 THEN QS$ = QS$ + "0"
3040 QS$ = QS$ +  STR$ (QD(1)) + " ON "
       +  STR$ (QD(3)) + "." +
       MO$(QD(4)) + "." +  STR$ (QD(5))
3042   IF  NOT QU% THEN  PRINT YD$
3062   GOSUB 9800: FOR I = 0 TO QN: PRINT
       QS$(I): NEXT I
3080 I =  FRE (0)
3100   GOSUB 9990:CV = 2: POKE 34,2:J =
       INT (1 + RM + (T / TM))
3105   GOSUB 7000: IF GD% < 0 THEN 3160
------------------------------------------------------------------------
                                                 READS TS# DISC FILE
3145   PRINT
       D4$"OPENTS#/"X$",L40";VS$(3):
       PRINT D4$"READTS#/"X$",R"RI:
       INPUT A$: PRINT D4$"CLOSE"
3150 T%(RI) = T%(RI) + 10: IF QU% THEN
       T%(RI) = 13
3160 QS$ = B$(CY)
```

```
--------------------------------------------------------------------------
                                                   ADDS THE SUCCESS ENDING
                                                   TO THE DECISION STRING
3170  IF GD% < 0 THEN 3190
3172  IF EQ(1,0) AND MS% THEN T%(RI) =
       13:QI = 910 +  INT ( RND (1) *
       8): GOSUB 9505:A$ = QS$  GOTO
       3190
3180 QI =  VAL (A$):QT% = QU%  GOSUB
       9505:QT% = QU%
3190  GOSUB 9700:QT% = 0
3191 B$(0) = QS$
3200  PRINT :QI = 901 + T2  GOSUB 9500:
       GOTO 3240
3210  PRINT :QI = 900: GOSUB 9500
--------------------------------------------------------------------------
                                                   FUTURE DECISION START
3240  PRINT :QI = 904  GOSUB 9500: GET
       A$: PRINT A$: IF A$ = "N" THEN
       3320
3260  IF A$ <  > "Y" THEN 3240
3280 CY = CY + 1
3282  GOSUB 30: GOTO 1280
3300 QS$ = Y$: GOSUB 9800: PRINT : PRINT
      "YOU ARE PLANNING TO";  IF UQ%
      THEN  PRINT " REQUEST: ": GOTO
      3310
3305  PRINT ":"
3310  FOR I = 0 TO QN: PRINT QS$(I):
      NEXT I:B$(CY) = U$: GOTO 3210
3320 UU$ = "N": IF DN < 2 THEN 3440
3325  GOSUB 30:UU$ = U$
--------------------------------------------------------------------------
                                                   PREVIOUS DECISION LINKING
3330  PRINT : HOME :QI = 902 + T2: GOSUB
       9500
3340  GOSUB 7600
3360  IF  VAL (QS$) = 0 AND QS$ <  > "0"
      THEN  PRINT "ERR": GOTO 3340
3380  GOSUB 9100: FOR I = 0 TO QN: IF
      QV(I) AND QV(I) > DN - 1 THEN
      PRINT "ERROR": GOTO 3340
3400  NEXT I:A$ = QS$
3440  GOSUB 30: PRINT :QI = 903: GOSUB
       9500
3460  GOSUB 7600
3480  IF  VAL (QS$) = 0 AND QS$ <  > "0"
      THEN  PRINT "ERR": GOTO 3460
3500  GOSUB 9100: FOR I = 0 TO QN: IF
      QV(I) AND QV(I) > MC THEN  PRINT
      "ERROR". GOTO 3440
3520  NEXT I
3540  GOSUB 7400
--------------------------------------------------------------------------
                                                   END OF DECISION PROCEDURE
3541  PRINT : HOME : PRINT "THANK YOU.
      YOUR DECISION NUMBERED "DN
3542  IF QU% THEN  PRINT "TO REQUEST
      INFORMATION HAS BEEN": PRINT
      "SUBMITTED": GOTO 3560
3550  PRINT "IS BEING TRANSMITTED TO THE
      APPROPRIATE UNITS."
3560  PRINT
      D4$"OPENR#",CD$",L500"VS$(4):
      PRINT D4$"WRITER#"CD$",R"RI
3580  PRINT B$(0):B$(0) = LT$: PRINT RM:
      PRINT CP: PRINT MC: PRINT DN:
      PRINT CY: FOR I = 0 TO CY: PRINT
      B$(I):B$(I) = "": PRINT
```

```
          R$(I):R$(I) = "": NEXT I: PRINT
          UU$: PRINT A$: PRINT U$: PRINT _
          QS$: PRINT D4$"CLOSE"
3590    GOSUB 9950
3600    PRINT :QI = 905: GOSUB 9500: POKE
          SB%,0:I = DL
3605    IF ( PEEK (KB%) < 128) AND (I > 0)
          THEN I = I - 1: GOTO 3605
3609    POKE SB%,0
3610    PRINT : HOME :SD% = 1  GOTO 600
---------------------------------------------------------------------
                                            ALLOWS EXPERIMENTER TO
                                            ENTER MESSAGES IN INTERACTIVE
                                            MODE
7000 PK% =  - 1:GD% =  - 1: FOR I = 0 TO
          211
7002    IF T%(I) < 0 THEN 7050
7005    IF PK% >  - 1 AND (GD% >  - 1 OR
          ((I > 210) OR (J > 210))) THEN
          7055
7010    IF PK% >  - 1 THEN 7030
7020    IF ((T%(I) < 1 OR T%(I) > 3) OR I
          < J) AND (T%(I) < 10) THEN PK% =
          I
7030    IF GD% >  - 1 THEN 7050
7040    IF (T%(I) > 0 AND T%(I) < 4) AND
          (I >  = J) THEN GD% = I
7050    NEXT I: PRINT 7050: END
7055 RI = GD%
7060    IF GD% >  - 1 THEN 7090
7070 RI = PK%
7080 T%(RI) =  - T%(RI): IF  NOT T%(RI)
          THEN T%(RI) =  - 10
7090    RETURN
7100 I = 211: IF SM$ <  > "" THEN 7130
7110    FOR I =  INT (RM) + 1 TO 210:QI =
          ABS (T%(I)): IF (QI = 4 OR QI =
          5) AND (I >  = PM) THEN 7130
7120    NEXT I
7130    PRINT D4$"IN#2": POKE SW% + 1,0:
          HOME : PRINT "NEXT TIME="I"
          ENTER TIME:";: INPUT I
7131    IF I < 0 THEN  END
7132    IF I = 211 THEN  PRINT "": GOTO
          7160
7134    INPUT "ENTER MESSAGE:":SM$
7135    IF SM$ = "" THEN 7160
7150 T%(I) =  - 16
7160    HOME : PRINT D4$"IN#0": RETURN
7400    IF DN >  = KD% AND CP >  = KP%
          THEN T2 = 6
7410    RETURN
---------------------------------------------------------------------
                                            PROMPTS AND ACCEPTS THE
                                            DECISION ENTERED BY
                                            PARTICIPANT
7500    IF BB >  - 1 THEN  FLASH : PRINT
          "SELECT";: NORMAL : PRINT " 1 TO
          "M;: IF (BB > 0) AND ( LEN (F$) >
          3) THEN  PRINT " OR *";
7501    IF BB >  - 1 THEN  PRINT ":";
7510    POKE SB%,0: GOTO 7550
7520    HTAB 1: FLASH : PRINT "PLEASE
          HURRY-SELECT";: NORMAL :
          PRINT " 1 TO "M":";
7550 QI = DZ
7552    IF BB < 0 THEN S$ = "1": GOTO 7599
7555 QJ =  PEEK (KB%)
7560    IF (QJ < 128) AND (QI > 0) THEN QI
          = QI - 1: GOTO 7555
7570    IF QI <  = 0 THEN 7520
7580 S$ =  CHR$ (QJ - 128)
7582    IF (BB > 0) AND ((S$ = "*") AND (
          LEN (F$) > 3)) THEN BB =  - 1:S$
          = "1": PRINT "* = COMPLETE ENTRY"
7590    IF BB >  = 0 THEN  HTAB 1: PRINT
          "ENTERED:"S$"
          "
```

```
7599  POKE SB%.0: RETURN
----------------------------------------------------------------------
                                             KEYBOARD CHARACTER ACCEPT
                                             UTILITY ROUTINE
7600 QS$ = ""
7620 QI =  POS (0): FLASH : PRINT " ";:
       HTAB QI + 1: NORMAL
7630  POKE SB%,0
7640 QJ =  PEEK (KB%): IF QJ < 128 THEN
       7640
7650 QJ = QJ - 128
7660  IF QJ = 13 THEN  PRINT " ": GOTO
       7799
7665  PRINT  CHR$ (QJ);
7670  IF (QJ > 47) AND (QJ < 58) THEN
       7700
7680  IF (QJ < 48) AND (QJ > 43) THEN QJ
       = 59
7690  IF QJ = 59 THEN 7700
7691 QS$ = "ERR": PRINT : GOTO 7799
7700 QS$ = QS$ +  CHR$ (QJ): GOTO 7620
7799  POKE SB%.0: RETURN
----------------------------------------------------------------------
                        SET SUBROUTINE
----------------------------------------------------------------------
                                             DIMENSION VARIABLES
8000  DIM
       MO$(12).OM%(8,128).T%(211).U%(200)

8001  DIM OM(12),VS$(16)
----------------------------------------------------------------------
                                             INITIALIZE VARIABLES
8004 MM$ = "IF YOU WISH TO MAKE A
       DECISION. HIT THE 'D' KEY"
8005 AA$(0) = "ACTION AREA:".YD$ = "YOU
       HAVE MADE THE DECISION TO
       ":AA$(1) = "FUTURE " + AA$(0)
8009 DL = 2400:DZ = 4000
8010 LS% = 10
8011 D4$ =  CHR$ (4):P0$ =  CHR$ (13) +
       D4$ + "PR#0":PR$ = P0$
8012  FOR I = 1 TO 12: READ MO$(I): NEXT
       I: DATA
       JAN.FEB,MAR,APR,MAY,JUN,JUL,AUG,SE
       P,OCT,NOV.DEC
8013  FOR I = 0 TO 5: READ QS(I): NEXT
       I: DATA 60,60,24,28,12,0
8020 DH$ =
       "-----------------------------------
       ------"
8022 KB% =  - 16384:SB% =  - 16368:B0% =
       - 16287:VT = 99999:SW% =  - 16296
8023  IF  PEEK (B0%) > 127 THEN B0% =
       2048
----------------------------------------------------------------------
                                             EXPERIMENTER INFORMATION
                                             ENTRY
8024  INPUT "ENTER D#,P#:";KD%,KP%
8025 T2 = 6 * ((KD% < 0) AND KP% < 0)
8032  PRINT "ALL MOVES SUCCESSFUL ?
       (Y/N):";: GET A$: PRINT A$: IF A$
       = "N" THEN 8037
8034  IF A$ <  > "Y" THEN 8032
8036 MS% = 1
8037  PRINT : PRINT "PRINTER ?
       (P,G,N):";: GET A$: PRINT A$: IF
       (A$ <  > "P") AND (A$ <  > "G")
       AND (A$ <  > "N") THEN 8037
8038  IF A$ = "N" THEN 8042
```

```
8039 PR$ = D4$ + "PR#1" +  CHR$ (13)
8040  IF A$ = "G" THEN PR$ = PR$ +  CHR$
       (9) + "20L"
8042  INPUT "ENTER PARTICIPANT
       CODE:";A$:CD$ = "/" + A$: INPUT
       "ENTER SCENARIO:";X$: PRINT
       D4$"VERIFYV/"X$: PRINT
       D4$"READV/"X$: FOR I = 0 TO 16
8043  GET A$: IF  ASC (A$) = 13 THEN
       8045
8044 VS$(I) = VS$(I) + A$: GOTO 8043
8045  NEXT I: PRINT D4$: PRINT
       D4$"CLOSE"
8050  PRINT : PRINT "DISPLAY MINUTE
       MARKERS? (Y/N):";: GET A$: PRINT
       A$: IF (A$ = "") OR ((A$ <  >
       "N") AND (A$ <  > "Y")) THEN 8050
8052  IF A$ = "Y" THEN DM% = 1
8053  PRINT
--------------------------------------------------------------------------------------------
                                                           READS DATA FILES FOR
                                                           PROGRAM CONTROL
8059  PRINT D4$"VERIFYLOC/"X$;VS$(2):
       PRINT D4$"READLOC/"X$
8060  FOR I = 0 TO 5: INPUT DT(I): NEXT
       I: INPUT TM: INPUT SD
8070  FOR I = 0 TO IS: FOR J = 0 TO 2:
       INPUT OM%(J,I): NEXT J: NEXT I
8080  FOR I = 1 TO IS: IF OM%(1,I) >  =
       0 THEN 8090
8081  INPUT QS$: GOSUB 9200:OM%(1,I) = X
       + OM%(1,0):OM%(2,I) = Y +
       OM%(2,0)
8090  NEXT I: PRINT D4$"CLOSE"
8092  PRINT "NEW OR RESTART? (N/R):";:
       GET A$: PRINT A$
8093  IF A$ = "N" THEN 8500
8094  IF A$ <  > "R" THEN 8092
8110  PRINT D4$"VERIFYR";CD$;VS$(4):
       PRINT D4$"READR";CD$: INPUT
       RM,CP,MC,DN:RM =  INT (RM)
8112  PRINT "TIME="RM" P#="CP" M#="MC"
       D#"DN:
8115 PM = RM
8120 J = RM * 60 * TM:I = 29030400:QA =
       5: GOSUB 9900:I = 2419200: GOSUB
       9900
8130 I = 86400: GOSUB 9900:I = 3600:
       GOSUB 9900
8140 I = 60: GOSUB 9900:I = .99: GOSUB
       9900
8150  GOTO 8510
8500  PRINT D4$"VERIFY TM/"X$;VS$(3)
8501  PRINT D4$"READ TM/"X$:PM =  - 1
8510  INPUT TN: FOR I = 0 TO TN: INPUT
       T%(I)
8518  IF T%(I) > 999 THEN U%(T%(I) -
       1000) = 1
8520  NEXT I: PRINT D4$"CLOSE"
8521 TN = 210
8530  GOSUB 9630
8700  IF RM THEN 8999
8701  PRINT : PRINT
       D4$"OPENATBL/"X$;VS$(5)",L"SZ:
       PRINT D4$"READATBL/"X$",R1"
8702  INPUT J: PRINT D4$"CLOSE"
8703  PRINT D4$"OPENA"CD$;VS$(4)",L"SZ
8704  PRINT D4$"WRITEA"CD$",R1"
8705  PRINT J
```

```
8706  PRINT D4$"CLOSE"
8709  FOR I = 2 TO J
8710  PRINT
        D4$"OPENATBL/"X$;VS$(5)",L"SZ
8720  PRINT D4$"READATBL/"X$",R"I
8730  INPUT A$
8740  PRINT D4$"CLOSE"
8750  PRINT D4$"OPENA"CD$;VS$(4)",L"SZ
8760  PRINT D4$"WRITEA"CD$",R"I
8770  PRINT A$
8780  PRINT D4$"CLOSE"
8799  NEXT I
----------------------------------------------------------------------
                                                PRINT INSTRUCTIONS TO
                                                PARTICIPANT
8800  FOR I = 921 TO 921:QI = I: HOME :
        GOSUB 9500:QI = 905: GOSUB 9500:
        GET A$: NEXT I
8999  RETURN
----------------------------------------------------------------------
                                                SPLITS CHARACTER STRINGS
                                                INTO SMALLER STRINGS
9100 QW = 0:QO = 0:QN = 0:QA = 0:QL =
        LEN (QS$):QV(1) = 0:QE = QL:QV(0)
        =  VAL (QS$): FOR QI = 2 TO QL:
        IF  MID$ (QS$,QI,1) <  > ";" THEN
        9130
9110 QN = QN + 1:QI = QI + 1:QV(QN + 1)
        = 0:QV(QN) =  VAL ( MID$
        (QS$,QI,99))
9120  GOTO 9190
9130  IF  MID$ (QS$,QI,1) <  > "#" THEN
        9160
9140 QI = QI + 1:QO =  VAL ( MID$
        (QS$,QI,99))
9150  GOTO 9190
9160  IF  MID$ (QS$,QI,1) <  > "&" THEN
        9180
9170 QI = QI + 1:QW = 1:QW$ =  MID$
        (QS$,QI,99)
9175  GOTO 9190
9180  IF  MID$ (QS$,QI,1) <  > "@" THEN
        9199
9181  FOR JZ = QL TO QI + 1 STEP  - 1:
        IF  MID$ (QS$,JZ,1) = "@" THEN
        QA$ =  MID$ (QS$,QI,JZ - QI +
        1):QA = QI:QI = JZ + 1: GOTO 9185
9183  NEXT JZ: PRINT 9183: END
9185  IF UQ% THEN QU%(M) =  VAL ( MID$
        (QS$,QA + 1,99))
9186  IF QE > QA - 1 THEN QE = QA - 1:
        RETURN
9190  IF QE > QI - 2 THEN QE = QI - 2
9199  NEXT QI: RETURN
----------------------------------------------------------------------
                                                CONVERTS TEXT XY STRINGS
                                                INTO NUMBERS FOR DISTANCE
                                                COMPUTATION
9200 X = ( ASC ( LEFT$ (QS$,1)) - 65) *
        26 +  ASC ( MID$ (QS$,2,1)) -
        65:Y =  VAL ( MID$ (QS$,3,99))
9210  RETURN
----------------------------------------------------------------------
                                                STRING PRINOUT ROUTINES
9399 D4$ =  CHR$ (4):QI = 900
9400  PRINT D4$"READM"QI
9401  PRINT 9401: END
9410 A$ = ""
9412  GET QQ$: IF QQ$ =  CHR$ (13) THEN
```

```
      9415
9413 A$ = A$ + QQ$: GOTO 9412
9415  IF  LEFT$ (A$,1) = "]" THEN  PRINT
       : GOTO 9499
9420 QB = 1:QN =  VAL (A$): IF QN THEN
       QB = 2
9422  IF QN < 0 THEN QB = 3:QN =  - QN:
       PRINT
9423  PRINT
9425  IF QN = 1 THEN  INVERSE
9426  IF QN = 2 THEN  FLASH
9430  PRINT  MID$ (A$,QB,255);
9440  NORMAL : GOTO 9410
9499  PRINT D4$"CLOSE": RETURN
-----------------------------------------------------------------------------
                                                     CONVERTS BINARY FILES
                                                     FROM APPLEWRITER INTO
                                                     APPLESOFT CHARACTER STRINGS
9500 QZ = 0:QB = 1: GOTO 9510
9504 QZ = 1:QB = 1: GOTO 9510
9505 QZ = 1:QB = 2
9510  PRINT : PRINT
       D4$"BLOADTEXT.M"QI;VS$(1)",A"LT:QL
       =  PEEK (43616) +  PEEK (43617) *
       256 - 2 - QZ
9515  IF QZ THEN QQ = QL: GOTO 9555
9520 QN = QB + 39: IF QN > QL THEN QN =
       QL
9525 QQ = QI
9530  IF  PEEK (LT + QB) = 141 THEN
       PRINT :QB = QB + 1: GOTO 9520
9535  FOR QI = QN TO QB STEP  - 1:QV =
       PEEK (QI + LT)
9540  IF QV = 141 THEN QQ = QI: GOTO
       9555
9545  IF (QV = 224) OR ((QV = 32) OR (QV
       = 96)) THEN QQ = QI - 1: GOTO
       9555
9550  NEXT QI
9555  FOR QJ = QB TO QQ:QA =  PEEK (QJ +
       LT): IF QA < 64 THEN  INVERSE :QA
       = QA + 64
9556  IF QT% AND QA <  > 192 THEN 9575
9557 QT% = 0
9560  I: QA > 223 THEN QA = QA - 64
9561  IF QA > 128 THEN QA = QA - 128
9565  IF  NOT QZ THEN 9570
9566  IF QA <  > 13 THEN QS$ = QS$ +
       CHR$ (QA)
9567  GOTO 9571
9570  PRINT  CHR$ (QA);
9571  NORMAL : IF QA = 13 THEN 9577
9575  NORMAL : NEXT QJ
9577 QB = QJ + 1: IF QZ THEN  RETURN
9580 QV =  PEEK (LT + QB): IF (QV = 224)
       OR ((QV = 32) OR (QV = 96)) THEN
       QB = QB + 1: GOTO 9580
9585  IF QB < QL THEN  PRINT : GOTO 9520
9590  RETURN
-----------------------------------------------------------------------------
                                                     PRINTS A STRING
9600  GOSUB 9800: FOR J = 0 TO QN: PRINT
       QS$(J): NEXT J: PRINT : RETURN
-----------------------------------------------------------------------------
                                                     WAITS FOR KEY HIT
9610  POKE SB%,0: PRINT : INVERSE :
       PRINT "HIT ANY KEY TO CONTINUE":
       GET QS$(9): NORMAL : PRINT :
       RETURN
```

```
-----------------------------------------------------------------------------------------
                                                    PRINTS A MESSAGE
9620  FLASH :MC = MC + 1: PRINT ""DH$:
       NORMAL : PRINT "MESSAGE
       NUMBER="MC: PRINT T$: PRINT
9621  RETURN
-----------------------------------------------------------------------------------------
                                                    PRINTS A DECISION
9630 PM = PM + 1: VTAB 1: HTAB 36: IF
       DM% THEN  PRINT PM;
9631  VTAB CV + 2: HTAB 1: IF T%(PM) = 0
       OR T%(PM) =  - 10 THEN 9630
9640  RETURN
-----------------------------------------------------------------------------------------
                                                    REQUEST FOR INFORMATION
                                                    AND ACCOUNT UPDATING
9650  PRINT PR$: PRINT : PRINT : INVERSE
       : PRINT ""DH$: NORMAL : PRINT
       :DN = DN + 1: PRINT "DECISION
       NUMBER="DN: PRINT T$
9651  RETURN
9660 QL =  LEN (QS$): FOR I = 1 TO QL:
       IF  MID$ (QS$,I,1) = "@" THEN
       9670
9662  NEXT I:SS$ = "": RETURN
9670 SS$ =  RIGHT$ (QS$,QL - I + 1):QS$
       =  LEFT$ (QS$,I - 1): RETURN
9690 MG = 1: GOTO 9701
9700 MG = 0
9701 QL =  LEN (SS$): IF QL < 2 THEN
       RETURN
9702  GOSUB 10000:I = 0:QI = 1
9704  IF  MID$ (SS$,QI,1) <  > "@" THEN
       9750
9705  GOSUB 9770:A$ = "":J = 0:QI = QI +
       1: IF  MID$ (SS$,QI,1) = "!" THEN
       J = 1
9706 QI = QI + 1: FOR K = QI TO QL: IF
       MID$ (SS$,K,1) = ">" THEN 9708
9707  NEXT K
9708  IF  MID$ (SS$,K + 1,1) >  = "A"
       THEN 9745
9709 I =  ASC ( MID$ (SS$,QI,1)): IF I <
       65 OR I > 74 THEN 9712
9710 A$ = AC$(I - 65):I =  ASC ( LEFT$
       (A$,1)): GOTO 9715
9712 A$ =  MID$ (SS$,QI,K - QI): IF
       RIGHT$ (A$,1) = "Q" THEN A$ = "="
       + QQ$
9715 QM =  VAL ( MID$ (A$,2,9)):QK =
       VAL ( MID$ (SS$,K + 1,254)): IF
       NOT QM THEN 9780
9716  IF MG THEN 9723
9717  IF J THEN 9725
9718  IF (T%(RI) = 11 OR T%(RI) = 12)
       AND  NOT QT% THEN 9745
9719 QS$ = QS$ + "@!" + A$ + ">" +  STR$
       (QK) + "@"
9721  GOTO 9745
9723  IF (T%(RI) = 11) OR (T%(RI) = 12)
       AND QK THEN 9745
9724  GOTO 9730
9725  IF T%(RI) <  > 13 THEN 9745
9730  PRINT D4$"OPENA"CD$",L"SZ;VS$(4):
       PRINT D4$"READA"CD$",R"QM
9731  INPUT LI$: PRINT D4$"CLOSE": IF I
       <  > 61 THEN 9735
9733  IF QK THEN  PRINT
       D4$"OPENA"CD$",L"SZ;VS$(4): PRINT
```

```
           D4$"WRITEA"CD$",R"QK: PRINT LI$:
           PRINT D4$"CLOSE": GOTO 9745
9734     PRINT "THE ANSWER IS:": GOSUB
           9790: PRINT : GOTO 9745
9735 LI =  VAL (LI$)
9736     PRINT D4$"OPENA"CD$",L"SZ;VS$(4):
          · PRINT D4$"READA"CD$",R"QK: INPUT
           LO$: PRINT D4$"CLOSE":LO =  VAL
           (LO$)
9737     IF I = 45 THEN LO = LO - LI
9738     IF I = 42 THEN LO = LO * LI
9739     IF I = 43 THEN LO = LO + LI
9744 LI$ =  STR$ (LO): GOTO 9733
9745 QI = QN
9750 QI = QI + 1: IF QI < QL THEN 9704
9760     RETURN
9770     FOR QN = QI + 1 TO QL: IF  MID$
           (SS$,QN,1) = "@" THEN 9779
9771     NEXT QN
9779     RETURN
9780     IF  NOT J THEN 9719
9781 LI$ =  MID$ (A$,2,6): GOTO 9733
9790 QS$ = LI$:ZN = QN: GOSUB 9800: FOR
           I = 0 TO QN: PRINT QS$(I): NEXT
           I:QN = ZN: RETURN
9800 QB = 1:QS$(0) = QS$:QN = 0:QL =
           LEN (QS$): IF QL < 40 THEN
           RETURN
------------------------------------------------------------
                                              SPLITS UP STRING INTO
                                              SMALLER STRINGS
9805 QN =  - 1
9820     FOR QI = QB + 38 TO QB STEP  - 1
9830     IF  MID$ (QS$,QI,1) = " " THEN
           9850
9840     NEXT QI: PRINT "END9840": END
9850 QN = QN + 1:QS$(QN) =  MID$
           (QS$,QB,QI - QB + 1):QB = QI + 1
9855     IF QL - QB > 39 THEN 9820
9856 GN = QN + 1:QS$(QN) =  RIGHT$
           (QS$,QL - QB + 1)
9860     RETURN
9900 K =  INT (J / I):DT(QA) = DT(QA) +
           K:J = J - K * I:QA = QA - 1:
           RETURN
------------------------------------------------------------
                                              STORES RESULT FILES ON DISK
9920     PRINT : PRINT
           D4$"OPENR#"CD$;VS$(4)",L500":
           PRINT D4$"READR"CD$",R"PM: INPUT
           QS$: PRINT D4$"CLOSE": IF PR THEN
           PRINT PR$
9950     PRINT D4$"OPENR";CD$;VS$(4): PRINT
           D4$"WRITER";CD$: PRINT RM: PRINT
           CP: PRINT MC: PRINT DN: PRINT TN:
           FOR I = 0 TO 210: PRINT T%(I):
           NEXT I: PRINT D4$"CLOSE": RETURN
9990 CV =  PEEK (37): POKE 34,CV: FOR QI
           = 1 TO LS%: PRINT : NEXT QI:
           PRINT PO$: RETURN
------------------------------------------------------------
                                              SPLITS STRING INTO SMALLER
                                              STRINGS
10000    FOR I = 1 TO QL
10010    IF  MID$ (SS$,I,1) <  > ")" THEN
           10100
10020 K =  ASC ( MID$ (SS$,I + 1,1)) -
           65: IF K < 0 OR K > 9 THEN 10100
10030    FOR J = I - 1 TO 1 STEP  - 1: IF
```

```
        MID$ (SS$,J,1) = "@" THEN 10040
10035   NEXT J
10040  J = J + 2:AC$(K) =  MID$ (SS$,J,I
        - J)
10050   IF  RIGHT$ (AC$(K),1) = "Q" THEN
        AC$(K) =  LEFT$ (AC$(K),1) + QQ$
10100   NEXT I: RETURN
-------------------------------------------------------------------------------
                                                   MISCELLANEOUS UTILITY
                                                   ROUTINES, NOT EXECUTED
                                                   BY SIM.
15000  I = 61:D4$ =  CHR$ (4):CD$ =
        "/CLIFF":QM = 1: GOTO 9730
30000   INPUT B$: PRINT B$
50000  I$ =  CHR$ (9):Q$ =  CHR$ (27):D$
        =  CHR$ (4):S$ =  CHR$ (31):M$ =
        CHR$ (30):L$ =  CHR$ (29):NC$ =
        CHR$ (2):EX$ =  CHR$ (1)
50002   PRINT D$"PR#0"
50005   PRINT D$"PR#1"
50006   PRINT Q$"J,0,960,$"
50007   PRINT Q$"B,6,$"
50010   PRINT I$0"N"
50020   PRINT Q$"R,2,$"
50030   PRINT M$NC$
50100   END
55000  D$ =  CHR$ (4): PRINT D$"OPEN
        ADDLIST": PRINT D$"WRITE
        ADDLIST": LIST : PRINT D$"CLOSE":
        END
56000   INPUT S: PRINT S: END
```

KEY VARIABLES

CD$ = /PARTICIPANT NAME
X$ = SCENARIO NAME
QS$ = DECISION ALTERNATIVE CHOSEN BY PARTICIPANT
B$(CY)= INFORMATION SEARCH REQUESTED BY PARTICIPANT
OD,OH,I = DAYS, HOURS, MINUTES REQUIRED TO IMPLEMENT DECISION
DN = DECISION NUMBER
SM$ = MESSAGE ENTERED BY EXPERIMENTER DURING INTERACTIVE SESSIONS
I = TIME SLOT NUMBER
MC = MESSAGE NUMBER
T$ = MESSAGE STRING

VEDIT PROGRAM

7.1 GENERAL INFORMATION

The VEDIT Program is used to define the locations of the various programs used in the simulation.

7.2 USING VEDIT

7.2.1 RUN the VEDIT program.

7.2.2 Enter the input scenario to be edited.
If new, enter "new."

7.2.3 The current volume assignments for the
16 file groups are shown.

7.3 DEFINITION OF FILE GROUPS

FILE GROUPS must be put onto a single disk volume. There are 17 of these file groups as follows:

0	VEDIT; V/s; SIM; DEDIT
1	TEDITOR, TEXT.Mn
2	LEDIT: LOC/s
3	TEDIT: TS#t/s; TM/s
4	Rt/p; A/p
5	AEDIT, ATBL/s
6	MEASURE
7	D
8-16	D1-D9

7.4 DATA ENTRY

The entry defines the location of the slot number, disk number and volume number. Sample entry: ,s6,d1,v254. Note that the entry must begin with a comma. This particular entry would be correct for the normal floppy disk number 1.

7.5 TERMINATING THE PROGRAM

7.5.1 Terminate with a Q command.

7.5.2 Enter the scenario name. The V/s file will be written on the volume specified by FILE GROUP 0.

7.6 FILE FORMATS

7.6.1 INPUT FILE

NAME: V/s
where s is the scenario name
SOURCE: VEDIT
EXAMPLE: V/YUGOSLAV DILEMMA

7.6.2 OUTPUT FILE

NAME: V/s
where s is the scenario name
SOURCE: VEDIT
EXAMPLE: V/YUGOSLAV DILEMMA

7.7 PROGRAM LISTING

VEDIT

```
                                                     COMMENTS
-----------------------------------------------------------------------------
                                                     START OF PROGRAM
10  REM  VEDIT
11  TEXT : HOME
50 D4$ =  CHR$ (4)
55  GOSUB 8000
60  GOTO 1800
100  REM
-----------------------------------------------------------------------------
                                                     PRINT SCREEN AND ALLOWS
                                                     USER TO SPECIFY THE VOLUME,
                                                     SLOT, AND DRIVE NUMBER FOR
                                                     EACH OF THE PROGRAMS AND
                                                     FILES LISTED
110   HOME : FOR I = 0 TO 16: PRINT
        I"="VS$(I)" -> "TT$(I)
112   FOR J = 1 TO  LEN (VS$(I)): IF
        MID$ (VS$(I),J,1) < "," THEN 115
113   NEXT J: GOTO 190
115 VS$(I) =  LEFT$ (VS$(I),J - 1)
190   NEXT I
200   VTAB 19: PRINT : PRINT "HIT E TO
        EDIT,Q TO QUIT:";
210   GET A$: PRINT A$
212   IF A$ = "Q" THEN 900
215   IF A$ <  > "E" THEN 110
220   INPUT "ENTER LINE NUMBER:";L
230   IF L < 0 OR L > 16 THEN  PRINT
        "ERR": GOTO 220
300   VTAB 19: CALL  - 958: PRINT : PRINT
        "ENTER DATA/"L" (,S#,D#,V#):";
310 A$ = "":NV = 0:NS = 0:NC = 0:ND = 0
320   GET B$:A =  ASC (B$): PRINT B$;
321   IF A = 13 THEN 335
322   IF B$ = "," THEN NC = NC + 1
323   IF B$ = "V" THEN NV = NV + 1
324   IF B$ = "D" THEN ND = ND + 1
325   IF B$ = "S" THEN NS = NS + 1
332 A$ = A$ + B$
333   GOTO 320
-----------------------------------------------------------------------------
                                                     ERROR MESSAGE
335   IF (NC <  > 3) OR (NV <  > 1) OR
        (ND <  > 1) OR (NS <  > 1) OR (
        LEFT$ (A$,1) <  > ",") THEN
        PRINT "ERR": GOTO 300
339 VS$(L) = A$
340   GOTO 110
-----------------------------------------------------------------------------
                                                     WRITES A NEW V/SCENARIO
                                                     DISK FILE
900   INPUT "ENTER OUTPUT SCENARIO NAME
        OR RETURN FORNO OUTPUT FILE:";F$
910   IF F$ = "" THEN  END
999   PRINT D4$"OPENV/"F$VS$(0): PRINT
        D4$"WRITEV/"F$: FOR I = 0 TO 16:
        PRINT VS$(I): NEXT I: PRINT
        D4$"CLOSE": END
1200   PRINT D4$"PR#1": GOTO 200
1500   PRINT D4$"PR#0": GOTO 200
1800   PRINT D4$"CLOSE"
1805   ONERR  GOTO 1850
-----------------------------------------------------------------------------
                                                     READS EXISTING V/SCENARIO
                                                     DISK FILE
1810   INPUT "ENTER INPUT SCENARIO
        NAME:";F$
```

```
1820   IF F$ = "" THEN 1800
1830   PRINT D4$"OPENV/"F$
1835   PRINT D4$"READV/"F$: FOR I = 0 TO
        16
1836 A$ = ""
1837   GET B$:A =  ASC (B$): IF A = 13
        THEN 1840
1839 A$ = A$ + B$: GOTO 1837
1840 VS$(I) = A$: NEXT I: PRINT D4$
1849   GOTO 1890
1850   INPUT "NEW SCENARIO? (Y/N) ";AN$
1851   IF AN$ = "N" THEN 1800
1852   IF AN$ <  > "Y" THEN 1800
1855   POKE 216,0
1890   PRINT D4$"CLOSE"
1899   POKE 216,0: GOTO 100
1910   RETURN
-----------------------------------------------------------------
                                              DIMENSIONS VARIABLES AND
                                              DISPLAYS PROGRAMS AND FILES
8000   DIM VS$(16),TT$(16)
8010   FOR I = 0 TO 16
8020   READ TT$(I): NEXT I
8030   DATA V/S; VEDIT; SIM; DEDIT,TEXT.;
       TEDITOR;,LOC/S; LEDIT,TM/S;
        TS#N/S; TEDIT,RN/P; A/P
8035   DATA ATBL/S;
        AEDIT,MEASURE,D,D1,D2,D3,D4,D5,D6,
        D7,D8,D9
8999   RETURN
-----------------------------------------------------------------
                                              UTILITY ROUTINES NOT
                                              EXECUTED BY VEDIT
30000   INPUT B$: PRINT B$
50000 I$ =  CHR$ (9):Q$ =  CHR$ (27):D$
        =  CHR$ (4):S$ =  CHR$ (31):M$ =
        CHR$ (30):L$ =  CHR$ (29):NC$ =
        CHR$ (2):EX$ =  CHR$ (1)
50002   PRINT D$"PR#0"
50005   PRINT D$"PR#1"
50006   PRINT Q$"J,0,960,$"
50007   PRINT Q$"B,6,$"
50010   PRINT I$0"N"
50020   PRINT Q$"R,2,$"
50030   PRINT M$NC$
50100   END
55000 D$ =  CHR$ (4): PRINT D$"OPEN
        ADDLIST": PRINT D$"WRITE
        ADDLIST": LIST : PRINT D$"CLOSE":
        END
```

KEY VARIABLES

F$ = SCENARIO NAME

TT$(16)= NAMES OF PROGRAMS AND FILES THAT APPEAR ON SCREEN

B$ = USER DEFINED VOLUME, SLOT, AND DRIVE NUMBER

L = LINE NUMBER ON THE SCREEN THAT IS TO BE EDITED

PROFILE (MEASURE) PROGRAM

8.1 GENERAL

The PROFILE program (formerly called the MEASURE program) analyzes participant responses.

8.2 USING PROFILE

Criswell, Unger, Swezey and Streufert (1983) provide details for operating the PROFILE program.

8.3 FILES

8.3.1 INPUT FILES

NAME: Rt/p
where t is the time, p is
participant code
SOURCE: SIM
EXAMPLE: R5/JOHN DOE
CONTENTS: Message

Real Time Minutes

For each cycle including
Message cycle:
Real Time of Day
Number of Decision Cycles

Real Time of Day Previous Decisions

Real Time of Day Previous Messages

NAME: R/p
where p is the participant code
SOURCE: SIM
EXAMPLE: R/JOHN DOE
CONTENTS: Input File generated in SIM in
the same format as the TM/s.

8.4 PROGRAM LISTING

PROFILE

```
                                                     COMMENTS
------------------------------------------------------------------------
                                          INITIALIZATION
100 D4$ =  CHR$ (4): PRINT D4$"PR#1":
       GOTO 200
150  REM  1/5/84
200  TEXT :CV = 2:CS =  - 1:D4$ =  CHR$
       (4)
250  HOME : VTAB 10: HTAB 10: PRINT
       "PROGRAM MEASURE"
300  GOSUB 9900
------------------------------------------------------------------------
                                          READ PARTICIPANT'S DATA FILES
                                          AND COMPUTES MAIN DATA LIST
350 LP%(0) =  - 1
400  FOR I = 0 TO 9:MC%(0,I) = 1000:
       NEXT I
450  FOR I = 0 TO TN
500 J = T%(I): IF J < 10 THEN 600
550 NM = NM + 1:ML%(NM) = I
600  IF  ABS (J) < 10 OR  ABS (J) > 21
       THEN 1550
650  PRINT D4$"OPENR#"CD$",L500"
700  PRINT D4$"READR#"CD$",R"I
750 TD = TD + 1
800  INPUT ML$,RM,ZP,ZM,ZD,CY
------------------------------------------------------------------------
                                          COMPUTE MEASURE 1
850 ZP = ZP + 1:M1(ZP) = M1(ZP) + 1:K =
       ZD:UD%(K) = 0
900 RM = RM + .05:RM =  INT (RM * 10) /
       10:RM(K) = RM
950 NF%(K) = CY: IF ZD < MC%(0,ZP) THEN
       MC%(0,ZP) = ZD
1000  IF ZD > MC%(1,ZP) THEN MC%(1,ZP) =
       ZD
1050  FOR L = 0 TO CY: INPUT
       FT$(L),FD$(L,K):FD%(L,K) =  VAL (
       MID$ (FD$(L,K),3,3)): NEXT L
1100  INPUT
       PD$(K,0),PD$(K,1),AM$(K,0),AM$(K,1
       ): PRINT D4$"CLOSE"
1150  IF  NOT P THEN 1550
1200  PRINT "R"I;CD$: PRINT ML$: PRINT
       "TIME="RM: PRINT "PERIOD="ZP"
       MESSAGES="ZM
------------------------------------------------------------------------
                                          PRINTS OUT MAIN DATA LIST
1250  PRINT "DECISION NUMBER="ZD"
       TIME="FT$(0)
1300  PRINT "("FD$(0,K)")": IF  NOT CY
       THEN 1400
1350  PRINT "FUTURE DECISIONS:";: FOR II
       = 1 TO CY: PRINT
       "("FD$(II,K)")";: NEXT II: PRINT
1400  PRINT "BASED ON
       DECISIONS:"PD$(K,1)
1450  PRINT "BASED ON MESSAGES:"AM$(K,1)
1500  PRINT
1550  FOR L = 0 TO CY:FT$(L) = "": NEXT
       L:PD$(K,0) = "":AM$(K,0) = "":
       NEXT I
------------------------------------------------------------------------
                                          COMPUTES VECTOR LISTING
                                          FOR EACH PERIOD
1600 MC = TD
1650  PRINT
1700  FOR I = 1 TO CP:SS = 0:N%(I) =
       N%(I - 1)
```

```
1710  IF MC%(0,I) ) MC%(1,I) THEN 4400
1750  FOR J = MC%(0,I) TO MC%(1,I)
1800  IF  NOT  VAL (AM$(J,1)) THEN 1950
1850 QS$ = AM$(J,1): GOSUB 11300  FOR II
       = 0 TO QN:KK = QV(II):N%(I) =
       N%(I) + 1:K = N%(I):D%(K,0) =
       RM(J):D%(K,1) = ML%(KK):RT(K) =
       RM(J)
1900 D%(K,3) =  - KK:D%(K,2) = J:D%(K,4)
       = FD%(0,J):D%(K,5) = D%(K,4):
       GOSUB 9600  NEXT II
1950  IF  NOT M3(I) THEN 2100
2000  FOR K = 1 TO M3(I):  IF FD%(0,J) =
       TY%(K,I) THEN 2200
2050  NEXT K
----------------------------------------------------------------------------------
                                                          COMPUTE MEASURE 3
2100 M3(I) = M3(I) + 1
2150 TY%(M3(I),I) = FD%(0,J)
2200  NEXT J
2250  FOR J = MC%(0,I) TO MC%(1,I)
2300  IF NF%(J) ( 1 THEN 3250
2350  FOR K = 1 TO NF%(J)
2400 PV = 0
2450  IF (J + 1) ) MC THEN 3100
2500  FOR II = J + 1 TO MC
2550  IF FD%(K,J) (  ) FD%(0,II) THEN
       3050
2600 QS$ = PD$(II,1): GOSUB 11300
2650 JJ = 0
2700  IF QV(JJ) (  ) J THEN 3000
2750 N%(I) = N%(I) + 1:M = N%(I):D%(M,0)
       = RM(J):D%(M,1) = RM(II):D%(M,2)
       = J:D%(M,3) = II:D%(M,4) =
       FD%(0,D%(M,2)):D%(M,5) =
       FD%(0,D%(M,3))
2800 PV = 1: GOSUB 9600
2850  IF FD%(0,D%(M,3)) = FD%(0,D%(M,2))
       THEN D%(M,2) =  - J: GOSUB 9600:
       GOTO 2950
2900 UD%(J) = 1:UD%(II) = 1
2950  GOTO 3050
3000 JJ = JJ + 1: IF JJ (  = QN THEN
       2700
3050  NEXT II
3100  IF PV THEN 3200
3150 N%(I) = N%(I) + 1:M = N%(I):D%(M,0)
       = RM(J):D%(M,1) = RM(J):D%(M,2) =
       J:D%(M,3) = J:D%(M,4) =
       FD%(0,D%(M,2)):D%(M,5) =
       FD%(K,J): GOSUB 9600
3200  NEXT K
3250  NEXT J
3300  FOR J = MC%(0,I) TO MC%(1,I)
3350 QS$ = PD$(J,1): GOSUB 11300: IF
       NOT QV(0) THEN 3900
3400  FOR K = 0 TO QN
3450  IF NF%(QV(K)) ( 1 THEN 3650
3500  FOR II = 1 TO NF%(QV(K))
3550  IF FD%(II,QV(K)) = FD%(0,J) THEN
       3850
3600  NEXT II
3650 N%(I) = N%(I) + 1:M = N%(I):D%(M,0)
       = RM(J):D%(M,1) =
       RM(QV(K)):D%(M,2) = J:D%(M,3) =
       QV(K)::D%(M,4) =
       FD%(0,D%(M,2)):D%(M,5) =
       FD%(0,D%(M,3))
3700  GOSUB 9600:UD%(QV(K)) = 1:UD%(J) =
          1
3750   IF FD%(0.D%(M.3)) = FD%(0.D%(M.2))
         THEN D%(M.2) =  - J
3800   GOSUB 9600
3850   NEXT K
3900   NEXT J
3910  X9(I) = 0
3950   FOR J = MC%(0.I) TO MC%(1.I)
4000   IF UD%(J) THEN 4150
4050  N%(I) = N%(I) + 1:M = N%(I) D%(M.0)
         = RM(J) D%(M.1) = RM(J) D%(M.2) =
         J D%(M.3) = 0 D%(M.4) =
         FD%(0.D%(M.2)) D%(M.5) = 0
4100   GOSUB 9600
4150   NEXT J
4200   PRINT   PRINT "VECTORS FOR PERIOD=
         "I: PRINT
4250   FOR K = N%(I - 1) + 1 TO N%(I):
         FOR J = 0 TO 5. PRINT D%(K.J)"
         ".. NEXT J: PRINT   NEXT K
4300   PRINT
4350   NEXT I
----------------------------------------------------------------------
                                              COMPUTES MEASURE 14
4360   FOR I = 1 TO CP
4365   FOR J = MC%(0.I) TO MC%(1.I)
4370  X9(I) = X9(I) +  NOT UD%(J)
4380   NEXT J: NEXT I
----------------------------------------------------------------------
                                              COMPUTES NUMBER OF CATEGORIES
                                              AND PRINTS.
4400  NV = 0:VT%(0) = D%(0.4)
4450   FOR K = 0 TO N%(CP): IF ((D%(K.0)
         = D%(K.1)) AND (D%(K.2) =
         D%(K.3))) THEN 4750
4500   FOR J = 4 TO 5:L = D%(K.J): FOR M
         = 0 TO NV: IF (VT%(M) = L) OR
         NOT L THEN 4700
4550   IF VT%(M) ( L THEN 4650
4600   FOR QI = NV TO M STEP  - 1:VT%(QI
         + 1) = VT%(QI): NEXT QI:NV = NV +
         1:VT%(M) = L: GOTO 4700
4650   NEXT M:NV = NV + 1:VT%(NV) = L
4700   NEXT J
4750   NEXT K: PRINT : PRINT "NUMBER OF
         CATEGORIES= "NV + 1: FOR K = 0 TO
         NV: PRINT VT%(K): NEXT K
----------------------------------------------------------------------
                                              COMPUTE MEASURES
4800   FOR I = 1 TO CP
4850   FOR J = N%(I - 1) + 1 TO N%(I)
----------------------------------------------------------------------
                                              COMPUTE MEASURE 13
4900   IF ((D%(J.0) = D%(J.1)) AND
         (D%(J.2) = D%(J.3)) AND (D%(J.5))
         AND (D%(J.2) ) 0)) THEN X3(I) =
         X3(I) + 1: GOTO 5500
4950   IF D%(J,3) )  = 0 THEN 5200
----------------------------------------------------------------------
                                              COMPUTE MEASURE 2 AND 11
5000  M2(I) = M2(I) + 1:QE(I) = QE(I) +
         RT(J) - D%(J,1)
5050   FOR M = J TO N%(I): IF (D%(M,1) (
         = D%(M.0)) OR (D%(M.2) ( 0) THEN
         5150
5100   IF D%(M.0) = D%(J.0) THEN 5500
----------------------------------------------------------------------
                                              COMPUTE MEASURE 8
5150   NEXT M:M8(I) = M8(I) + 1: GOTO
```

```
        5500
5200  IF D%(J,1) <  = D%(J,0) THEN 5400
---------------------------------------------------------------------------
                                                COMPUTE MEASURE 12
5250  IF D%(J,2) < 0 THEN X4(I) = X4(I)
        + 1: GOTO 5500
---------------------------------------------------------------------------
                                                COMPUTE MEASURES 4 AND 6
5300 M4(I) = M4(I) + 1:M6(I) = M6(I) +
        D%(J,1) - D%(J,0):F%(J) = 1
5350  GOTO 5500
5400  IF (D%(J,2) < 0) OR (D%(J,1) =
        D%(J,0)) THEN 5500
---------------------------------------------------------------------------
                                                COMPUTE MEASURE 7
5450 M7(I) = M7(I) + 1
5500  NEXT J
5501  NEXT I
5510  FOR I = 1 TO CP
5515 M5(I) = 0
5520  FOR J = N%(I - 1) + 1 TO N%(I)
5522  IF  NOT F%(J) THEN 5630
5525  GOSUB 9150
5526  GOSUB 13000
5630  NEXT J
5631  NEXT I
---------------------------------------------------------------------------
                                                PRINTS OUT THE 14 MEASURES
5640  FOR I = 1 TO CP
5650  PRINT : PRINT "PERIOD "I: PRINT
       "1-MEASURE="M1(I)MT$(1)
5660 DV = M1(I): IF  NOT DV THEN DV = 1
5700  PRINT : PRINT "2-MEASURE="M2(I)"
       " INT (M2(I) * 100 / DV)"%"MT$(2)
5750  PRINT : PRINT
       "3-MEASURE="M3(I)MT$(3)
5800  PRINT : PRINT "4-MEASURE="M4(I)"
       " INT (M4(I) * 100 / DV)"%"MT$(4)
5850  PRINT : PRINT "5-MEASURE="M5(I)"
       " INT (M5(I) * 100 / DV)"%"MT$(5)
5900  PRINT : PRINT "6-MEASURE="M6(I)"
       MINUTES"MT$(6)
5950  PRINT : PRINT "7-MEASURE="M7(I)"
       " INT (M7(I) * 100 / DV)"%"MT$(7)
6000  PRINT : PRINT "8-MEASURE="M8(I)"
       " INT (M8(I) * 100 / DV)"%"MT$(8)
6050  PRINT : PRINT "9-MEASURE="M9(I)MT$(9)
6100  PRINT : PRINT
       "10-MEASURE="M0(I)MT$(10)
6150  IF  NOT M2(I) THEN  PRINT : PRINT
       "11-MEASURE="0: GOTO 6250
6200  PRINT : PRINT "11-MEASURE="QE(I) /
       M2(I)MT$(11)
6250  PRINT : PRINT
       "12-MEASURE="X4(I)MT$(12)
6300  PRINT : PRINT
       "13-MEASURE="X3(I)MT$(13)
6350  PRINT : PRINT
       "14-MEASURE="X9(I)MT$(14)
6400  NEXT I
6450  GOTO 6750
6500  END
---------------------------------------------------------------------------
                                                READ DATA FILES
6550  PRINT D4$"OPENR"RI;CD$",D2": PRINT
       D4$"DELETE"RI;CD$: PRINT
       D4$"OPENR"RI;CD$: PRINT
       D4$"WRITER"RI;CD$
6600  INPUT M$: INPUT LT$: INPUT RM:
```

```
          INPUT CY: FOR I = 0 TO CY: INPUT
          B$(I): PRINT R$(I): NEXT I: INPUT
          UU$: INPUT A$. INPUT U$: INPUT
          QS$: PRINT D4$"CLOSE"
6650     GOSUB 12400
6700     PRINT :QI = 905: GOSUB 9500: GET
          A$: PRINT : HOME :SD% = 1: GOTO
          2450
------------------------------------------------------------------------------
                                               COMPUTES MEASURES 15 AND
                                               16
6750 NN = 0: FOR I = 1 TO MC:
6800 V =  VAL ( MID$ (FD$(0,I),3,3)): IF
          NOT NN THEN 7000
6850  FOR JJ = 1 TO NN
6900  IF V = CT%(JJ) THEN 7050
6950  NEXT JJ
7000 NN = NN + 1:JJ = NN:CT%(JJ) = V
7050 NT%(JJ) = NT%(JJ) + 1
7100  NEXT I
7150  PRINT
7200  PRINT "MEASURE 15-"MT$(15): PRINT
7250  FOR I = 1 TO NN
7300  FOR J = I TO NN
7350  IF NT%(J) <  = NT%(I) THEN 7500
7400 K% = CT%(J):CT%(J) = CT%(I):CT%(I)
      = K%
7450 K% = NT%(J):NT%(J) = NT%(I) NT%(I)
      = K%
7500  NEXT J
7550  NEXT I
7600  FOR I = 1 TO NN
7650  PRINT CT%(I),NT%(I)
7700  NEXT I
7750 N1 =  INT ((0.1 * MC) + 0.5):N5 =
          INT ((0 5 * MC) + 0.5)
7800 CA = 0:CB = 0:CD = 0:CE = 0
7850  FOR I = 1 TO NN:CA = CA + NT%(I)
7900  IF CA >  = N1 THEN 8000
7950  NEXT I
8000  FOR I = NN TO 1 STEP  - 1:CB = CB
       + NT%(I)
8050  IF CB >  = N1 THEN 8150
8100  NEXT I
8150  FOR I = 1 TO NN:CD = CD + NT%(I)
8200  IF CD >  = N5 THEN 8300
8250  NEXT I
8300  FOR I = NN TO 1 STEP  - 1:CE = CE
       + NT%(I)
8350  IF CE >  = N5 THEN 8450
8400  NEXT I
8450 MZ = 2 * (CA - CB) + (CD - CE)
8500  PRINT : PRINT : PRINT
       "15-MEASURE="MZ
8550  PRINT "16-MEASURE="MZ / NN;MT$(16)
8600  PRINT D4$"PR#0": END
------------------------------------------------------------------------------
                                               PROGRAM TERMINATES ON THE
                                               ABOVE LINE
------------------------------------------------------------------------------
                                               SUBROUTINE FOR COMPUTING
                                               MEASURE 9
9150 QI = 1:L = 0:QL = 0
9200  FOR M = 0 TO N%(CP): IF M = J THEN
         9450
9250  IF  NOT F%(M) THEN 9450
9350  IF (D%(M,0) = D%(J,0)) OR (D%(J,0)
       = D%(M,1)) THEN QI = QI + 1: GOTO
         9450
```

```
9400  IF (D%(M,1) = D%(J,1)) OR (D%(J,1)
       = D%(M,0)) THEN QI = QI + 1
---------------------------------------------------------------------
                                                  COMPUTE MEASURE 9
9450  NEXT M:M9(I) = (D%(J,1) - D%(J,0))
       * QI + M9(I)
9559  RETURN
9600  IF N%(I) < 1 THEN  RETURN
9650  FOR LK = 0 TO N%(I) - 1: FOR LV =
       0 TO 5
9700  IF D%(LK,LV) <   > D%(N%(I),LV)
       THEN 9800
9750  NEXT LV:N%(I) = N%(I) - 1: GOTO
       9850
9800  NEXT LK
9850  RETURN
---------------------------------------------------------------------
                          SETUP SUBROUTINE
---------------------------------------------------------------------

                                                  DIMENSIONS ARRAYS
9900 DIM MO$(12),T%(211),LD%(50,9),L$(40),T
      Y%(20,9),FD%(9,211),NF%(211),MC%(1
      ,9),RM(211),CT%(50),NT%(50),UD%(21
      1),RT(211)
9950  DIM OM(12),PD$(210,1),AM$(210,1),FD$(9
      ,210),ML%(210)
10000  DIM MT$(18),D%(210,5),QH%(20),VT%(200)
      ,F%(210),C%(210)
10040  DIM LC%(50)
---------------------------------------------------------------------
                                                  INITIALIZES VARIABLES
10050 N%(0) =  - 1
10100 D4$ =  CHR$ (4)
10150  FOR I = 1 TO 12: READ MO$(I):
       NEXT I: DATA
       JAN,FEB,MAR,APR,MAY,JUN,JUL,AUG,SE
       P,OCT,NOV,DEC
10200  FOR I = 0 TO 5: READ QS(I): NEXT
       I: DATA 60,60,24,28,12,0
10250 DH$ =
       "-------------------------------
       ------"
10300 KB% =  - 16384:SB% =  - 16368
---------------------------------------------------------------------
                                                  READ PARTICIPANT NAME AND
                                                  READ DATA LIST. PRINT OUT
                                                  DATA LIST IF DESIRED
10350  INPUT "ENTER PARTICIPANT
       CODE:";A$
10400 CD$ = "/" + A$
10450  PRINT : PRINT "DATA LIST?
       (Y/N):";: GET A$: PRINT A$
10500  IF A$ <   > "Y" AND A$ <   > "N"
       THEN 10450
10550  IF A$ = "Y" THEN P = 1
10600  PRINT
10650  FOR I = 1 TO 16: READ MT$(I):
       NEXT I
---------------------------------------------------------------------
                                                  INTITIALIZES MEASURE NAMES
10700  DATA " (# OF DECISIONS)"," (# OF
       RESPONDENT DEC.)"," (# OF DEC.
       CATEGORIES)"," (# OF FWD
       INTEGRATIONS)"," (MULTIPLEXITY
       F)"," (WEIGHT)"," (# OF BKD
       INTEG)"
10750  DATA " (# OF UNINTEG.RES.DEC.)","
       (QIS)"," (WEIGHTED QIS)","
```

```
        (AVE RESPONSE SPEED)"
10800   DATA " (SERIAL CONNECTIONS)","
        (PLANNED INTEGRATIONS)"
10850   DATA " (GENERAL UNINTEGRATED
        DEC.)"
10900   DATA   " (SPREAD ACROSS
        DEC.CAT.)"   ," (AVERAGE SPREAD)"
10950   GOSUB 12400
------------------------------------------------------------------
                                            PRINT OUT INTIAL INFORMATION
11000   PRINT "NUMBER OF MINUTES IN
        SIMULATION "; INT (RM)
11050   PRINT "NUMBER OF MESSAGES=";CM
11100   PRINT "NUMBER OF DECISIONS=";CD
11150   PRINT "NUMBER OF PERIODS=",CP + 1
11200 CP = CP + 1
11250   RETURN

------------------------------------------------------------------
                                            SUBROUTINE FOR SPLITTING
                                            DATA STRINGS
11300 QW = 0:QO = 0:QN = 0:QL =  LEN
        (QS$):QV(1) = 0:QE = QL:QV(0) =
        VAL (QS$): FOR QI = 2 TO QL: IF
        MID$ (QS$,QI,1) (  > ";" THEN
        11400
11350 QN = QN + 1:QI = QI + 1:QV(QN + 1)
        = 0:QV(QN) =  VAL ( MID$
        (QS$,QI,99))
11400   IF QE > QI - 2 THEN QE = QI - 2
11450   NEXT QI: RETURN
11500 X = ( ASC ( LEFT$ (QS$,1)) - 65) *
        26 +  ASC ( MID$ (QS$,2,1)) -
        65:Y =  VAL ( MID$ (QS$,3,99))
11550   RETURN
------------------------------------------------------------------
                                            SUBROUTINE FOR CONVERTING
                                            BINARY (APPLEWRITER FILES)
                                            TO CHARACTER INFORMATION
11600 QD(QN) = DT(QN) + T
11650   FOR QB = QN TO 4:QI =  INT
        ((QD(QB) - (QB > 1)) / QS(QB))
11700 QD(QB) = QD(QB) - QI *
        QS(QB):QD(QB + 1) = DT(QB + 1) +
        QI: NEXT QB
11750   RETURN
11800   GOSUB 11900: FOR J = 0 TO QN:
        PRINT QS$(J): NEXT J: PRINT :
        RETURN
------------------------------------------------------------------
                                            WAIT SUBROUTINE
11850   POKE  - 16368,0: PRINT : INVERSE
        : PRINT "HIT ANY KEY TO
        CONTINUE": GET QS$(9): NORMAL :
        RETURN
------------------------------------------------------------------
                                            SUBROUTINE FOR LIMITING
                                            LENGTH OF TEXT LINES
11900 QB = 1:QS$(0) = QS$:QN = 0:QL =
        LEN (QS$): IF QL < 40 THEN
        RETURN
11950 QN =  - 1
12000   FOR QI = QB + 38 TO QB STEP  - 1
12050   IF  MID$ (QS$,QI,1) = " " THEN
        12150
12100   NEXT QI: PRINT "END9840": END
12150 QN = QN + 1:QS$(QN) =  MID$
        (QS$,QB,QI - QB + 1) QB = QI + 1
12200   IF QL - QB > 39 THEN 12000
```

```
12250 QN = QN + 1 QS$(QN) =  RIGHT$
       (QS$.QL - QB + 1)
12300  RETURN
12350 K =  INT (J / I):DT(QA) = DT(QA) +
       K J = J - K * I:QA = QA - 1:
       RETURN
-------------------------------------------------------------------
                                               SUBROUTINE FOR READING
                                               DATA FILES
12400  PRINT D4$"READR";CD$: INFUT
       RM.CP,CM,CD: INPUT TN: FOR I = 0
       TO TN: INPUT T%(I). NEXT I. PRINT
       D4$"CLOSE": RETURN
-------------------------------------------------------------------
                                               SUBROUTINE FOR COMPUTING
                                               MEASURES 5 AND 10
13000  REM  WEIGHTED QIS
13010 QC%(0) = 0:QC%(1) = 0:QC%(2) = 1
13090  FOR JK = 0 TO 1
13092  FOR II = 0 TO 210:C%(II) =  NOT
       F%(II): NEXT II:C%(J) = 1:LC%(0)
       = J
13095 PT = 0
13100  FOR JJ = N%(0) + 1 TO N%(CP)
13110  IF C%(JJ) THEN 13900
13115  FOR JL = 0 TO 1
13116 MF = 0
13120  IF (D%(JJ.JL) <  > D%(J,JK)) THEN
       13890
13130 QC%(JK) = QC%(JK) + 1:C%(JJ) = 1
13135  IF (JK) THEN QC%(2) = QC%(2) +
       1:MF = 1
13140 PT = 1.LC%(PT) = JJ
13190  FOR JM = N%(0) + 1 TO N%(CP)
13195  IF C%(JM) THEN 13400
13200  IF (D%(JM.JL) <  > D%(LC%(PT).(
       NOT JL))) THEN 13400
13210 PT = PT + 1:LC%(PT) = JM:QC%(JK) =
       QC%(JK) + 1
13215  IF MF AND  NOT JL THEN QC%(2) =
       QC%(2) + 1
13220 C%(JM) = 1
13230  GOTO 13190
13400  NEXT JM
13405  IF  NOT PT THEN 13890
13410 PT = PT - 1: IF PT THEN 13190
13890  NEXT JL
13900  NEXT JJ
13910  NEXT JK
13920 QI = 1 + QC%(0) + QC%(1)
-------------------------------------------------------------------
                                               COMPUTE MEASURE 10
13930 M0(I) = (D%(J,1) - D%(J.0)) * QI +
       M0(I)
-------------------------------------------------------------------
                                               COMPUTE MEASURE 5
13935 M5(I) = QC%(2) + M5(I)
13999  RETURN
-------------------------------------------------------------------
                                               MISCELLANEOUS UTILITY
                                               SUBROUTINES (NOT EXECUTED
                                               BY MEASURES PROGRAM)
50000 I$ =  CHR$ (9):Q$ =  CHR$ (27):D$
       =  CHR$ (4):S$ =  CHR$ (31):M$ =
       CHR$ (30):L$ =  CHR$ (29):NC$ =
       CHR$ (2):EX$ =  CHR$ (1)
50010  PRINT D$"PR#0"
50020  PRINT D$"PR#1"
50030  PRINT Q$"J,0,960,$"
```

```
50040   PRINT Q$"B,6,$"
50050   PRINT I$0"N"
50060   PRINT Q$"R,2,$"
50070   PRINT M$NC$
50080   END
50090  D$ =  CHR$ (4): PRINT D$"OPEN
        ADDLIST": PRINT D$"WRITE
        ADDLIST": LIST : PRINT D$"CLOSE":
        END
```

<u>KEY VARIABLES</u>

INITIAL DATA LIST VARIABLES
RM = NUMBER OF MINUTES IN SIMULATION
CM = NUMBER OF MESSAGES IN SIMULATION
CD = NUMBER OF DECISIONS IN SIMULATION
CP = NUMBER OF PERIODS IN SIMULATION
CD$ = /PARTICIPANT'S NAME

MAIN DATA LIST VARIABLES
ML$ = DECISION ALTERNATIVE SELECTED
RM = TIME IN SIMULATION WHEN DECISION WAS MADE (REAL MINUTES)
ZP = PERIOD IN WHICH DECISION WAS MADE
ZM = NUMBER OF MESSAGES THAT PRECEDED A DECISION
FT$(0) = REAL TIME WHEN DECISION WAS MADE
FD$(0,K) = DECISION ALTERNATIVE CODE
FD$(II,K) = FUTURE DECISION CODE
PO$(K,1) = DECISION NUMBERS OF PREVIOUS RELATED DECISIONS
AM$(K,1) = DECISION NUMBERS OF PREVIOUS RELATED MESSAGES
ZD = DECISION NUMBER

MANIPULATING KEY VARIABLES

Although the preceding section describes procedures for changing scenario variables, it seems appropriate to devote additional attention to the procedures required to manipulate key variables such as:

1) Time compression, decision charge time, and start time
2) Information load
3) Length of session
4) Ratio of responsive to fixed messages
5) Ratio of successful to unsuccessful messages

Further, a discussion of the system's capability to operate interactively is warranted.

Time compression, amount of time charged for each decision made by the participant, and simulation start time can be manipulated with the LEDIT program. After this program is run and the Yugoslav Dilemma scenario has been loaded (using the L command), the user should press the T key in order to inspect the time multiplier, charge time, and start time.

The value of the time multiplier (MUL on the screen) determines the number of simulation seconds that elapse for each real time second. The charge time (CHR on the screen), when multiplied by the MUL value, determines the amount of time that is charged for each decision that is made. Start time is displayed in day, month, and year. These values are easily changed by entering new values on the keyboard.

Information load, length of a session, ratio of responsive to fixed messages and successful to unsuccessful messages can all be manipulated through the TEDIT program. For all manipulations, run the TEDIT program, press L to load the scenario, then press E to edit.

Information load is manipulated by using the EDIT command of the TEDIT program. If more messages are required in a scenario, simply enter a message number at each point in time a message is to be added. Similarly, to reduce load, use the EDIT command to remove messages from a scenario.

To review the current sequence of messages by minute, select C (for cycle) from the Command Menu, and select the minutes desired for review. (For example, to review minutes 0 through 5, enter 0,5 when asked for seconds desired in the cycle. An example and explanation of a listing from minutes 0 to 14 follows:

SAMPLE	COMMENT
0>4:RANDOM (NOT CHECKED) MESSAGE 25 SOVIET AGENTS IN BULGARIA NEAR SOPHIA ARE TRAINING REBEL YUGOSLAV FORCES. BULGARIAN GOVERNMENT PROVIDING TRAIN- ING ASSISTANCE.	Minute 0 is a fixed message, message type 4. Message #25 appears.
1>0:NO MESSAGE	Minutes 1 and 2 have no message.
2>0:NO MESSAGE	
3>3:SUCCESS MESSAGE 423 3 HAS HAD THE FOLLOWING RESULT: 120 BULGARIAN MILITARY FORCES MOBILIZING ON BULGARIA/YUGOSLAV BORDER AND POSING THREAT TO YUGOSLAVIA.	Minute 3 may be a responsive message; if option is used, this message will be successful, message type 3. If no responsive message is due, fixed message #120 will appear.
4>0:NO MESSAGE	Minutes 4 and 5 have no message.
5>0:NO MESSAGE	
6>3:SUCCESS MESSAGE 423 3 HAS HAD THE FOLLOWING RESULT: 52 THE YUGOSLAV COMMUNITS HAVE PUBLI- CALLY CALLED FOR DOLANC'S RESIGNA- TION TO PUT AN END TO HIS AUTHORI- TARIAN PRACTICES.	Minute 6 is like Minute 3. If fixed message appears, it is message #52.
7>0:NO MESSAGE	Minutes 7 and 8 have no message.
8>0:NO MESSAGE	

```
             SAMPLE                                    COMMENT
9>3:SUCCESS MESSAGE 423                   Minute 9 is like Minute 3.
3 HAS HAD THE FOLLOWING RESULT:
11
THE KREMLIN INDICATES THAT THE
SOVIETS WILL INVADE YUGOSLAVIA IF
ANY MORE YUGOSLAV REBELS ARE
IMPRISONED.
----------------------------------
10>0:NO MESSAGE                           Minutes 10 and 11 have no message.
----------------------------------
11>0:NO MESSAGE
----------------------------------
12>4:RANDOM (NOT CHECKED) MESSAGE         Minute 12 has fixed message #17.
17
POWER PLANTS IN EASTERN MACEDONIA
HAVE BEEN SABOTAGED.  AREA WILL BE
WITHOUT ELECTRICITY FOR 2 DAYS.
----------------------------------
13>0:NO MESSAGE                           Minutes 13 and 14 have no message.
----------------------------------
14>0:NO MESSAGE
```

To change session length, simply change the time at which the scenario's break message ("This is the end of a period. You may now take a break...") and end message ("This is the end of the simulation.") appear. Presently, two break messages appear, one every 30 minutes, and the end message appears 30 minutes later. As an example, to change the length of the first period from 30 to 10 minutes, delete the break message (#70) from the minute 30 slot and place it in the minute 10 slot. With TEDIT in edit mode, enter 30 (for minute 30) when asked T? Enter 0 (from the menu) to delete the message from that slot. Next, with TEDIT in edit mode, enter 10 for minute 10 when asked T? Enter 7 (from the menu) for the break message type, then 70 for the real message number to be inserted in the minute 10 slot. The new array must then be saved (S on the command menu).

The current versions of the STORM and YUGOSLAV DILEMMA scenarios present two types of messages: fixed and responsive. Fixed messages are those that appear regardless of the decision alternatives executed by the participant. Responsive messages are related to the decisions executed by the participants. In order to vary the ratio of responsive to fixed messages, use the EDIT command of TEDIT to change the messages that are to appear during a run. Increasing the number of fixed messages requires entering the number of the message that is to appear at each point in time. Increasing the number of responsive messages is somewhat more complicated as there are two methods for presenting responsive messages. In their present form, the scenarios present specific responsive messages. These responsive messages are tied to decisions made by participants through account attachments on the end of decision string phrases (see Section 4.3.7). For example, if the decision is made to increase credit to Yugoslavia, the account attachment @?=45>0@ that appears at the end of the decision string will cause record number 45 of file A/PARTICIPANT to be printed as a responsive message at the first available slot in TEDIT (after the decision has been completed). To increase or modify this type of responsive message, one must change the coding at the end of decision alternatives, create text records in the AEDIT program, and indicate when the responsive messages are to appear in the TEDIT program. The latter task is accomplished by inserting a successful message ending (431) at each point in time where a responsive message is desired.

The second method for presenting responsive messages is less complicated but presents more general messages. This type of message simply repeats the decision that was executed and then indicates a successful, unsuccessful, or neutral outcome. To use this type of responsive message, the following steps must be taken:

1) Disable current responsive messages by deleting the @?=>0 code that appears at the end of DSP.

2) Create message endings using TEDITOR.

3) Insert these message endings (failure, success, neutral) into a scenario by using the EDIT command of the TEDIT program.

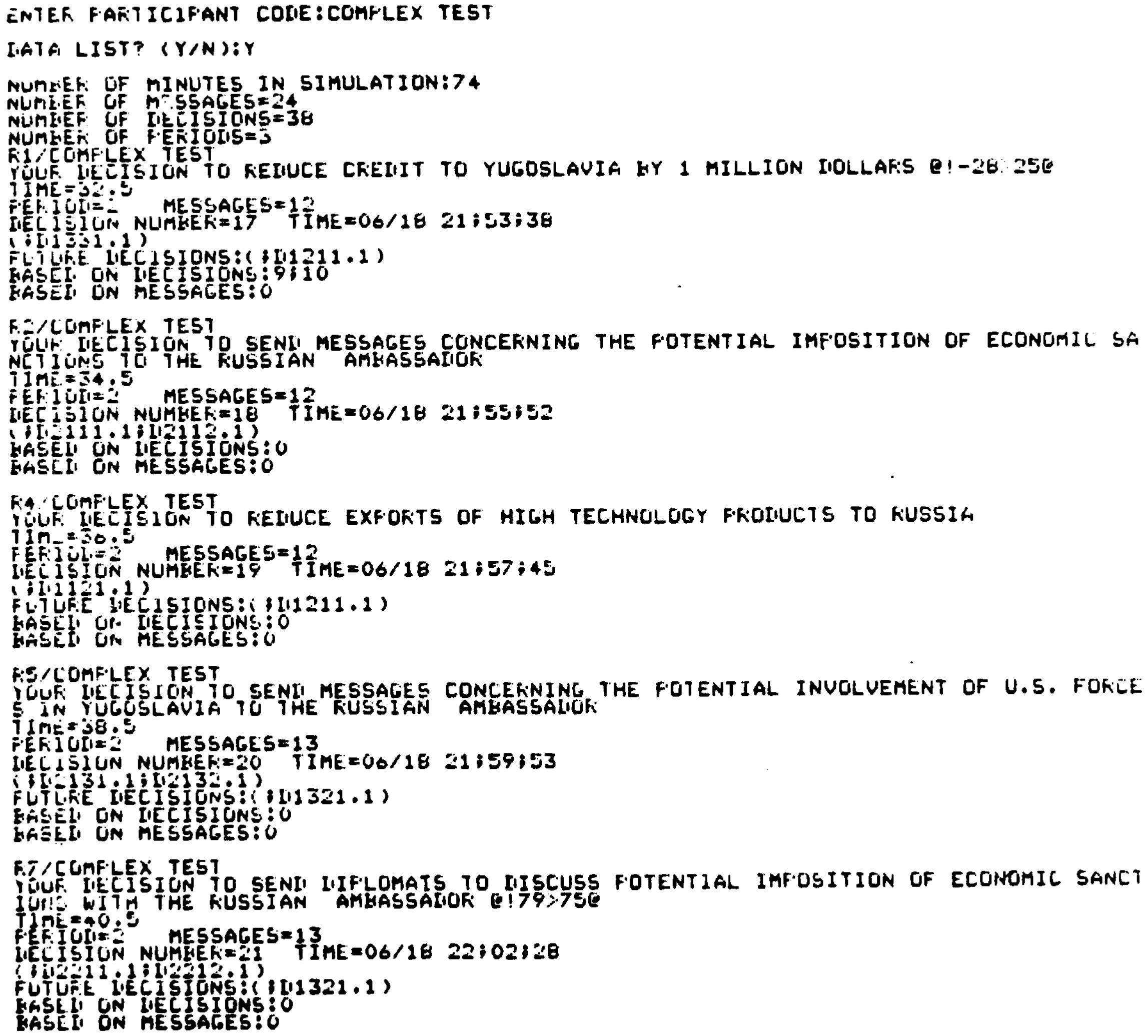
Interactive Mode

The current system does not operate interactively. That is, once a scenario has been set up, as described above, it will run as planned until it has ended. However, the system has been designed so that it can operate interactively if additional code is written for this purpose. In an interactive mode, the experimenter presses a button on a game paddle to indicate that he or she wants to enter a message. He or she then enters a message on the keyboard and the time in the simulation that it is to appear. This feature allows for more precise feedback and greater realism. Preliminary coding for the interactive mode occurs in lines 7000-7410 of the SIM program.

SCORING PARTICIPANTS' RESPONSES

During the course of the simulation all information relating to participants' decisions are recorded on the R#/PARTICIPANT NAME file. This text file is updated every time a decision is made. The PROFILE program reads this file in order to calculate the 14 measures.

An ⌊xample of a typical output from the PROFILE program is presented below:



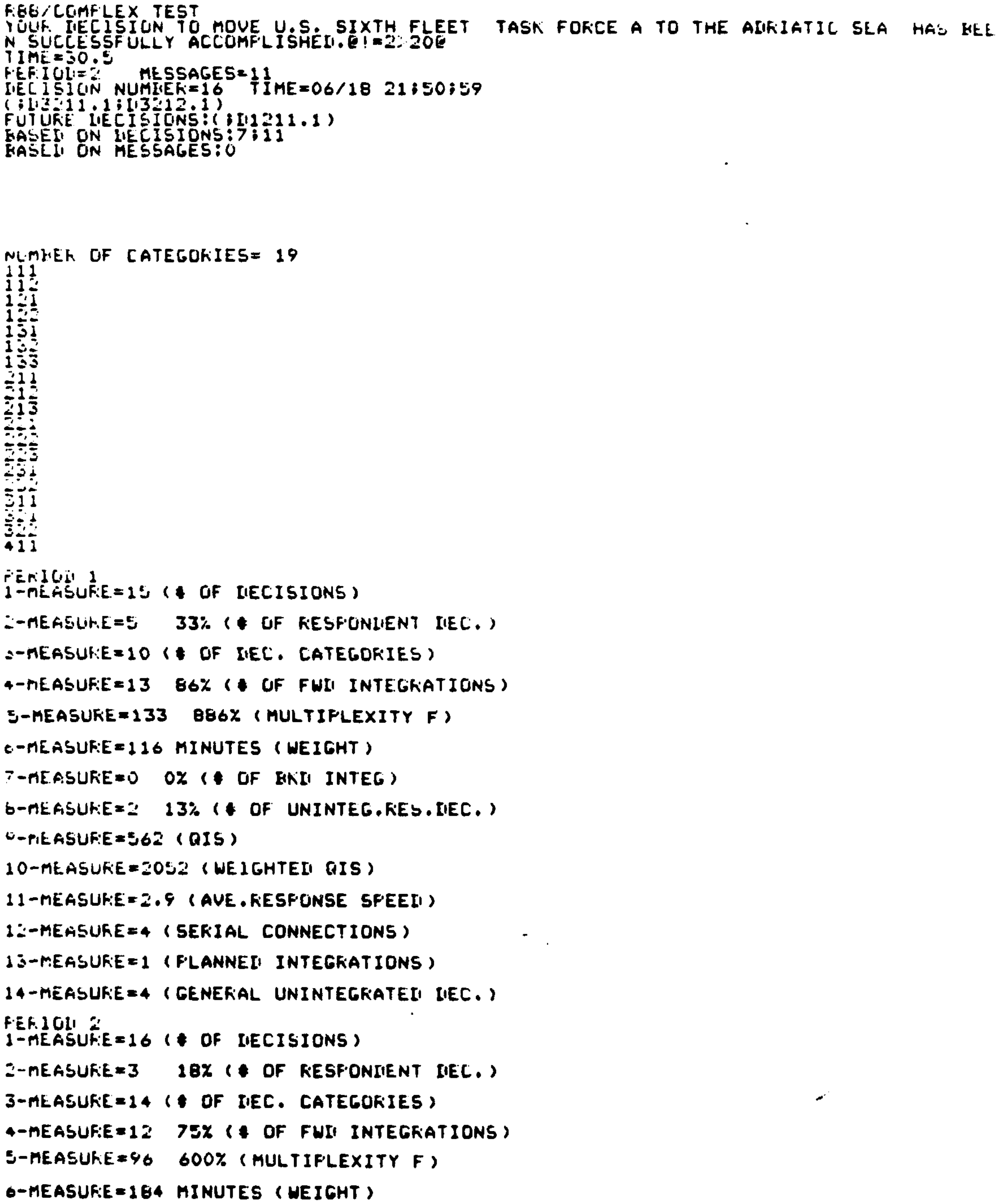
```
ENTER PARTICIPANT CODE:COMPLEX TEST

DATA LIST? (Y/N):Y

NUMBER OF MINUTES IN SIMULATION:74
NUMBER OF MESSAGES=24
NUMBER OF DECISIONS=38
NUMBER OF PERIODS=3
R1/COMPLEX TEST
YOUR DECISION TO REDUCE CREDIT TO YUGOSLAVIA BY 1 MILLION DOLLARS @!-28 250
TIME=32.5
PERIOD=2    MESSAGES=12
DECISION NUMBER=17  TIME=06/18 21;53;38
(;D1331.1)
FUTURE DECISIONS:(;D1211.1)
BASED ON DECISIONS:9;10
BASED ON MESSAGES:0

R2/COMPLEX TEST
YOUR DECISION TO SEND MESSAGES CONCERNING THE POTENTIAL IMPOSITION OF ECONOMIC SA
NCTIONS TO THE RUSSIAN  AMBASSADOR
TIME=34.5
PERIOD=2    MESSAGES=12
DECISION NUMBER=18  TIME=06/18 21;55;52
(;D2111.1;D2112.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:0

R4/COMPLEX TEST
YOUR DECISION TO REDUCE EXPORTS OF HIGH TECHNOLOGY PRODUCTS TO RUSSIA
TIM_=36.5
PERIOD=2    MESSAGES=12
DECISION NUMBER=19  TIME=06/18 21;57;45
(;D1121.1)
FUTURE DECISIONS:(;D1211.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:0

R5/COMPLEX TEST
YOUR DECISION TO SEND MESSAGES CONCERNING THE POTENTIAL INVOLVEMENT OF U.S. FORCE
S IN YUGOSLAVIA TO THE RUSSIAN  AMBASSADOR
TIME=38.5
PERIOD=2    MESSAGES=13
DECISION NUMBER=20  TIME=06/18 21;59;53
(;D2131.1;D2132.1)
FUTURE DECISIONS:(;D1321.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:0

R7/COMPLEX TEST
YOUR DECISION TO SEND DIPLOMATS TO DISCUSS POTENTIAL IMPOSITION OF ECONOMIC SANCT
IONS WITH THE RUSSIAN  AMBASSADOR @!79>75@
TIME=40.5
PERIOD=2    MESSAGES=13
DECISION NUMBER=21  TIME=06/18 22;02;28
(;D2211.1;D2212.1)
FUTURE DECISIONS:(;D1321.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:0
```

```
 R8/COMPLEX TEST
 YOUR DECISION TO SEND DIPLOMATS TO DISCUSS POTENTIAL RESUMPTION OF NORMAL TRADE W
 ITH THE RUSSIAN  AMBASSADOR @!=79>75@
 TIME=42.5
 PERIOD=2   MESSAGES=13
 DECISION NUMBER=22  TIME=06/18 22;04;50
 (;D2221.1;D2222.1)
 FUTURE DECISIONS:(;D1321.1)
 BASED ON DECISIONS:0
 BASED ON MESSAGES:0

 R9/COMPLEX TEST
 YOUR DECISION TO REDUCE EXPORTS OF FOOD TO RUSSIA  HAS BEEN SUCCESSFULLY COMPLETE
 D.
 TIME=.5
 PERIOD=1   MESSAGES=1
 DECISION NUMBER=1  TIME=06/18 19;40;38
 (;D1111.1)

FUTURE DECISIONS:(;D1121.1)
BASED ON DECISIONS:N
BASED ON MESSAGES:1

R10/COMPLEX TEST
YOUR DECISION TO REDUCE IMPORTS OF MANUFACTURED GOODS FROM RUSSIA
TIME=44.5
PERIOD=2   MESSAGES=13
DECISION NUMBER=23  TIME=06/18 22;07;12
(;D1221.1)
BASED ON DECISIONS:15;18
BASED ON MESSAGES:13

R11/COMPLEX TEST
YOUR DECISION TO ARRANGE A CONFERENCE WITH CABINET MEMBERS TO ASSESS PREVIOUS  PO
LITICAL ACTIONS @!=83>85@
TIME=46.5
PERIOD=2   MESSAGES=14
DECISION NUMBER=24  TIME=06/18 22;10;51
(;D2311.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:14

R13/COMPLEX TEST
YOUR DECISION TO SEND MESSAGES CONCERNING THE POTENTIAL IMPOSITION OF ECONOMIC SA
NCTIONS TO THE RUSSIAN  AMBASSADOR
TIME=48.5
PERIOD=2   MESSAGES=14
DECISION NUMBER=25  TIME=06/18 22;12;53
(;D2111.1;D2112.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:14

R14/COMPLEX TEST
YOUR DECISION TO ALERT U.S. SIXTH FLEET  TO PREPARE TO MOVE
TIME=50.5
PERIOD=2   MESSAGES=14
DECISION NUMBER=26  TIME=06/18 22;14;50
(;D3111.1)
FUTURE DECISIONS:(;D3221.1;D3222.1)
BASED ON DECISIONS:14
BASED ON MESSAGES:0

R15/COMPLEX TEST
YOUR DECISION TO REDUCE EXPORTS OF HIGH TECHNOLOGY PRODUCTS TO RUSSIA  HAS BEEN S
UCCESSFULLY COMPLETED.
TIME=2.5
PERIOD=1   MESSAGES=1
DECISION NUMBER=2  TIME=06/18 19;42;43
(;D1121.1)
FUTURE DECISIONS:(;D3211.1;D3212.1)
BASED ON DECISIONS:1
BASED ON MESSAGES:1

R16/COMPLEX TEST
YOUR DECISION TO REDUCE IMPORTS OF RAW MATERIALS FROM RUSSIA
TIME=52.5
PERIOD=2   MESSAGES=15
DECISION NUMBER=27  TIME=06/18 22;17;16
(;D1211.1)
FUTURE DECISIONS:(;D3221.1;D3222.1)
BASED ON DECISIONS:16;17;19
BASED ON MESSAGES:0

R17/COMPLEX TEST
YOUR DECISION TO REDUCE EXPORTS OF FOOD TO RUSSIA
TIME=54.5
PERIOD=2   MESSAGES=15
DECISION NUMBER=28  TIME=06/18 22;19;44
(;D1111.1)
FUTURE DECISIONS:(;D3221.1;D3222.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:0

R18/COMPLEX TEST
YOUR DECISION TO MOVE U.S. SIXTH FLEET  TASK FORCE A TO THE ADRIATIC SEA  HAS BEE
N SUCCESSFULLY ACCOMPLISHED.@!=2>20@
TIME=4.5
PERIOD=1   MESSAGES=2
DECISION NUMBER=3  TIME=06/18 19;45;03
```

```
(;D3211.1;D3212.1)
FUTURE DECISIONS:(;D1211.1)(;D3221.1;D3222.1)
BASED ON DECISIONS:2
BASED ON MESSAGES:0

R19/COMPLEX TEST
YOUR DECISION TO REDUCE CREDIT TO BULGARIA BY 1 MILLION DOLLARS @!-28 24@
TIME=56.5
PERIOD=2    MESSAGES=15
DECISION NUMBER=29  TIME=06/18 22;22;04
(;D1321.1)
FUTURE DECISIONS:(;D3221.1;D3222.1)
BASED ON DECISIONS:20;21;22
BASED ON MESSAGES:0

R20/COMPLEX TEST
YOUR DECISION TO ARRANGE A CONFERENCE WITH CABINET MEMBERS TO PLAN FUTURE  POLITI
CAL ACTIONS @!=84>86@
TIME=58.5
PERIOD=2    MESSAGES=15
DECISION NUMBER=30  TIME=06/18 22;24;40
(;D2321.1)
FUTURE DECISIONS:(;D3221.1;D3222.1)
BASED ON DECISIONS:14
BASED ON MESSAGES:0

R21/COMPLEX TEST
YOUR DECISION TO REDUCE IMPORTS OF RAW MATERIALS FROM RUSSIA  HAS BEEN SUCCESSFUL
LY COMPLETED.
TIME=6.5
PERIOD=1    MESSAGES=3
DECISION NUMBER=4  TIME=06/18 19;48;08
(;D1211.1)
FUTURE DECISIONS:(;D3221.1;D3222.1)
BASED ON DECISIONS:3
BASED ON MESSAGES:0

R22/COMPLEX TEST
YOUR DECISION TO REDUCE EXPORTS OF FOOD TO RUSSIA
TIME=60.5
PERIOD=2    MESSAGES=16
DECISION NUMBER=31  TIME=06/18 22;27;06
(;D1111.1)
FUTURE DECISIONS:(;D3221.1;D3222.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:0

R23/COMPLEX TEST
YOUR DECISION TO SEND MESSAGES CONCERNING THE POTENTIAL IMPOSITION OF ECONOMIC SA
NCTIONS TO THE RUSSIAN  AMBASSADOR
TIME=62.5
PERIOD=3    MESSAGES=18
DECISION NUMBER=32  TIME=06/19 00;01;50
(;D2111.1;D2112.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:18

R25/COMPLEX TEST
YOUR DECISION TO TRANSMIT FALSE INFORMATION ABOUT PLANNED US MILITARY ACTIONS IN
RUSSIA
TIME=64.5
PERIOD=3    MESSAGES=19
DECISION NUMBER=33  TIME=06/19 00;03;51
(;D4111.1)
FUTURE DECISIONS:(;D2231.1;D2232.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:19

R26/COMPLEX TEST
YOUR DECISION TO REDUCE EXPORTS OF HIGH TECHNOLOGY PRODUCTS TO RUSSIA
TIME=66.5
PERIOD=3    MESSAGES=20
DECISION NUMBER=34  TIME=06/19 00;06;26
(;D1121.1)
FUTURE DECISIONS:(;D1221.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:0

R27/COMPLEX TEST
YOUR DECISION TO MOVE U.S. AIR FORCE  INTERCEPTOR SQUADRONS (W. GERM) TO AIRFIELD
```

```
S IN BRITAIN @!=51/46@
TIME=66.5
PERIOD=3    MESSAGES=21
DECISION NUMBER=35  TIME=06/19 00;11;38
(;D3221.1;D3222.1)
FUTURE DECISIONS:(;D1211.1)(;D1331.1)(;D2231.1;D2232.1)
BASED ON DECISIONS:26;27;28;29;30;31
BASED ON MESSAGES:0

R29/COMPLEX TEST
YOUR DECISION TO REDUCE IMPORTS OF RAW MATERIALS FROM RUSSIA
TIME=70.5
PERIOD=3    MESSAGES=22
DECISION NUMBER=36  TIME=06/19 00;15;21
(;D1211.1)
FUTURE DECISIONS:(;D1111.1)
BASED ON DECISIONS:35
BASED ON MESSAGES:0

R31/COMPLEX TEST
YOUR DECISION TO REDUCE IMPORTS OF MANUFACTURED GOODS FROM RUSSIA  WAS NOT SUCCES
SFUL.
TIME=8.5
PERIOD=1    MESSAGES=3
DECISION NUMBER=5  TIME=06/18 19;50;48
(;D1221.1)
FUTURE DECISIONS:(;D3221.1;D3222.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:0

R32/COMPLEX TEST
YOUR DECISION TO REDUCE CREDIT TO YUGOSLAVIA BY 1 MILLION DOLLARS @!-28.25@
TIME=72.5
PERIOD=3    MESSAGES=23
DECISION NUMBER=37  TIME=06/19 00;17;25
(;D1331.1)
FUTURE DECISIONS:(;D1111.1)
BASED ON DECISIONS:35
BASED ON MESSAGES:0

R33/COMPLEX TEST
YOUR DECISION TO SEND DIPLOMATS TO DISCUSS POTENTIAL INVOLVEMENT OF U.S. FORCES I
N YUGOSLAVIA WITH THE RUSSIAN  AMBASSADOR @!=79/75@

TIME=74.5
PERIOD=3    MESSAGES=24
DECISION NUMBER=38  TIME=06/19 00;19;39
(;D2231.1;D2232.1)
BASED ON DECISIONS:33;35
BASED ON MESSAGES:0

R38/COMPLEX TEST
YOUR DECISION TO REDUCE EXPORTS OF HIGH TECHNOLOGY PRODUCTS TO RUSSIA  HAS BEEN S
UCCESSFULLY COMPLETED.
TIME=10.5
PERIOD=1    MESSAGES=4
DECISION NUMBER=6  TIME=06/18 19;53;25
(;D1121.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:0

R59/COMPLEX TEST
YOUR DECISION TO MOVE U.S. AIR FORCE  INTERCEPTOR SQUADRONS (W. GERM) TO AIRFIELD
S IN BRITAIN  HAS BEEN SUCCESSFULLY ACCOMPLISHED.@!
 =51/46@
TIME=12.5
PERIOD=1    MESSAGES=5
DECISION NUMBER=7  TIME=06/18 19;55;17
(;D3221.1;D3222.1)
FUTURE DECISIONS:(;D3211.1;D3212.1)(;D3221.1;D3222.1)(;D1311.1)(;D2121.1;D2122.1)

BASED ON DECISIONS:3;4;5
BASED ON MESSAGES:0

R52/COMPLEX TEST
YOUR DECISION TO MOVE U.S. AIR FORCE  INTERCEPTOR SQUADRONS (W. GERM) TO AIRFIELD
S IN BRITAIN  HAS BEEN SUCCESSFULLY ACCOMPLISHED.@!
 =51/46@
TIME=14.5
PERIOD=1    MESSAGES=5
```

```
DECISION NUMBER=8  TIME=06/18 19;59;59
(;D3221.1;D3222.1)
FUTURE DECISIONS:(;D3221.1;D3222.1)
BASED ON DECISIONS:7
BASED ON MESSAGES:5

R66/COMPLEX TEST
YOUR DECISION TO REDUCE CREDIT TO RUSSIA BY 1 MILLION DOLLARS  WAS NOT SUCCESSFUL
.
TIME=16.5
PERIOD=1    MESSAGES=6
DECISION NUMBER=9  TIME=06/18 20;04;05
(;D1311.1)
FUTURE DECISIONS:(;D1311.1)(;D1331.1)
BASED ON DECISIONS:7
BASED ON MESSAGES:5

R68/COMPLEX TEST
YOUR DECISION TO SEND MESSAGES CONCERNING THE POTENTIAL RESUMPTION OF NORMAL TRAD
E  TO THE RUSSIAN  AMBASSADOR  WAS NOT SUCCESSFUL.
TIME=18.5
PERIOD=1    MESSAGES=7
DECISION NUMBER=10  TIME=06/18 20;06;55
(;D2121.1;D2122.1)
FUTURE DECISIONS:(;D1331.1)(;D2211.1;D2212.1)
BASED ON DECISIONS:7
BASED ON MESSAGES:0

R70/COMPLEX TEST
YOUR DECISION TO MOVE U.S. AIR FORCE  INTERCEPTOR SQUADRONS (W. GERM) TO AIRFIELD
S IN BRITAIN  HAS BEEN SUCCESSFULLY AGCOMPLISHED.@!
 =51>46@
TIME=20.5
PERIOD=1    MESSAGES=7
DECISION NUMBER=11  TIME=06/18 20;09;53
(;D3221.1;D3222.1)
FUTURE DECISIONS:(;D3211.1;D3212.1)
BASED ON DECISIONS:8
BASED ON MESSAGES:0

R76/COMPLEX TEST
YOUR DECISION TO REDUCE CREDIT TO RUSSIA BY 1 MILLION DOLLARS  WAS NOT SUCCESSFUL
.
TIME=22.5
PERIOD=1    MESSAGES=8
DECISION NUMBER=12  TIME=06/18 20;12;29
(;D1311.1)
BASED ON DECISIONS:9
BASED ON MESSAGES:0

R78/COMPLEX TEST
YOUR DECISION TO REDUCE CREDIT TO BULGARIA BY 1 MILLION DOLLARS  HAS BEEN SUCCESS
FULLY COMPLETED.@!-28>24@
TIME=24.5
PERIOD=1    MESSAGES=9
DECISION NUMBER=13  TIME=06/18 20;14;25
(;D1321.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:0

R84/COMPLEX TEST
YOUR DECISION TO ARRANGE A CONFERENCE WITH CABINET MEMBERS TO PLAN FUTURE  POLITI
CAL ACTIONS  WAS NOT SUCCESSFUL.
TIME=26.5
PERIOD=1    MESSAGES=9
DECISION NUMBER=14  TIME=06/18 20;16;51
(;D2321.1)
FUTURE DECISIONS:(;D2321.1)(;D3111.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:0

R86/COMPLEX TEST
YOUR DECISION TO REDUCE EXPORTS OF FOOD TO RUSSIA  HAS BEEN SUCCESSFULLY COMPLETE
D.
TIME=26.5
PERIOD=1    MESSAGES=10
DECISION NUMBER=15  TIME=06/18 20;19;50
(;D1111.1)
BASED ON DECISIONS:0
BASED ON MESSAGES:9
```



```
R88/COMPLEX TEST
YOUR DECISION TO MOVE U.S. SIXTH FLEET  TASK FORCE A TO THE ADRIATIC SEA  HAS BEE
N SUCCESSFULLY ACCOMPLISHED.@!=2 20@
TIME=30.5
PERIOD=2    MESSAGES=11
DECISION NUMBER=16  TIME=06/18 21;50;59
(;D3211.1;D3212.1)
FUTURE DECISIONS:(;D1211.1)
BASED ON DECISIONS:7;11
BASED ON MESSAGES:0



NUMBER OF CATEGORIES= 19
111
112
121
122
131
132
133
211
212
213
221
222
223
231
232
311
321
322
411

PERIOD 1
1-MEASURE=15 (# OF DECISIONS)

2-MEASURE=5   33% (# OF RESPONDENT DEC.)

3-MEASURE=10 (# OF DEC. CATEGORIES)

4-MEASURE=13  86% (# OF FWD INTEGRATIONS)

5-MEASURE=133  886% (MULTIPLEXITY F)

6-MEASURE=116 MINUTES (WEIGHT)

7-MEASURE=0  0% (# OF BKD INTEG)

8-MEASURE=2  13% (# OF UNINTEG.RES.DEC.)

9-MEASURE=562 (QIS)

10-MEASURE=2052 (WEIGHTED QIS)

11-MEASURE=2.9 (AVE.RESPONSE SPEED)

12-MEASURE=4 (SERIAL CONNECTIONS)

13-MEASURE=1 (PLANNED INTEGRATIONS)

14-MEASURE=4 (GENERAL UNINTEGRATED DEC.)
PERIOD 2
1-MEASURE=16 (# OF DECISIONS)

2-MEASURE=3   18% (# OF RESPONDENT DEC.)

3-MEASURE=14 (# OF DEC. CATEGORIES)

4-MEASURE=12  75% (# OF FWD INTEGRATIONS)

5-MEASURE=96  600% (MULTIPLEXITY F)

6-MEASURE=184 MINUTES (WEIGHT)
```

```
7-MEASURE=2  12% (# OF BKD INTEG)
8-MEASURE=3  18% (# OF UNINTEG.RES.DEC.)
9-MEASURE=1312 (QIS)
10-MEASURE=3874 (WEIGHTED QIS)
11-MEASURE=3.83333333 (AVE.RESPONSE SPEED)
12-MEASURE=0 (SERIAL CONNECTIONS)
13-MEASURE=0 (PLANNED INTEGRATIONS)
14-MEASURE=2 (GENERAL UNINTEGRATED DEC.)
PERIOD 3
1-MEASURE=7 (# OF DECISIONS)
2-MEASURE=2   28% (# OF RESPONDENT DEC.)
3-MEASURE=7 (# OF DEC. CATEGORIES)
4-MEASURE=4  57% (# OF FWD INTEGRATIONS)
5-MEASURE=6  85% (MULTIPLEXITY F)
6-MEASURE=22 MINUTES (WEIGHT)
7-MEASURE=0  0% (# OF BKD INTEG)
8-MEASURE=1  14% (# OF UNINTEG.RES.DEC.)
9-MEASURE=134 (QIS)
10-MEASURE=612 (WEIGHTED QIS)
11-MEASURE=.5 (AVE.RESPONSE SPEED)
12-MEASURE=0 (SERIAL CONNECTIONS)
13-MEASURE=3 (PLANNED INTEGRATIONS)
14-MEASURE=2 (GENERAL UNINTEGRATED DEC.)
```

The information presented in the sample output above has been summarized in Table 1. Using the data in Table 1, a diagram called a time-event matrix was constructed and is presented in Figure 4. This matrix contains a point for each decision and clearly shows decision connections. The horizontal axis is time, the vertical axis is decision category. Forward integrations are noted by diagonal lines with a forward arrow ➔, backward integrations are diagonals with a backward arrow ←, serial connections are horizontal lines with a forward arrow ➔ .

Below is a detailed explanation of the calculation of each of the 14 measures. This explanation will refer to Table 1 and Figure 4. This explanation relies heavily on Appendix G of Criswell, Unger, Swezey and Streufert (1983).

TABLE 1. DATA STORED ON R#/PARTICIPANT NAME FILE

DECISION #	DECISION NUMBERS	BASED ON MESSAGE	FUTURE DECISIONS	PREVIOUS DECISIONS	TIME DECISION INITIATED
PERIOD 1:					
1	1111	1	1121	-	.5
2	1121	1	3211, 3212	1	2.5
3	3211	-	1211, 3221, 3222	2	4.5
4	1211	-	3221, 3222	3	6.5
5	1221	-	3221, 3222	-	8.5
6	1121	-	-	-	10.5
7	3221	-	3211, 3212, 3221, 3222, 1311, 2121, 2122	3, 4, 5	12.5
8	3221	5	3221, 3222	7	14.5
9	1311	5	1311, 1331	7	16.5
10	2121	-	1331, 2211, 2212	7	18.5
11	3221	-	3211, 3212	8	20.5
12	1311	-	-	9	22.5
13	1321	-	-	-	24.5
14	2321	-	2321, 3111	-	26.5
15	1111	9	-	-	28.5
PERIOD 2:					
16	3211	-	1211	7, 11	30.5
17	1331	-	1211	9, 10	32.5
18	2111	-	-	-	34.5
19	1121	-	1211	-	36.5
20	2131	-	1321	-	38.5
21	2211	-	1321	-	40.5
22	2221	-	1321	-	42.5
23	1221	13	-	15, 18	44.5
24	2311	14	-	-	46.5
25	2111	14	-	-	48.5
26	3111	0	3221, 3222	14	50.5
27	1211	-	3221, 3222	16, 17, 19	52.5
28	1111	-	3221, 3222	-	54.5
29	1321	-	3221, 3222	20, 21, 22	56.5
30	2321	-	3221, 3222	14	58.5
31	1111	-	3221, 3222	-	60.5
PERIOD 3:					
32	2111	18	-	-	62.5
33	4111	19	2231, 2232	-	64.5
34	1121	-	1221	-	66.5
35	3221	-	1211, 1331, 2231, 2232	26, 27, 28, 29, 30, 31	68.5
36	1211	-	1111	35	70.5
37	1331	-	1111	35	72.5
38	2231	-	-	33, 35	74.5

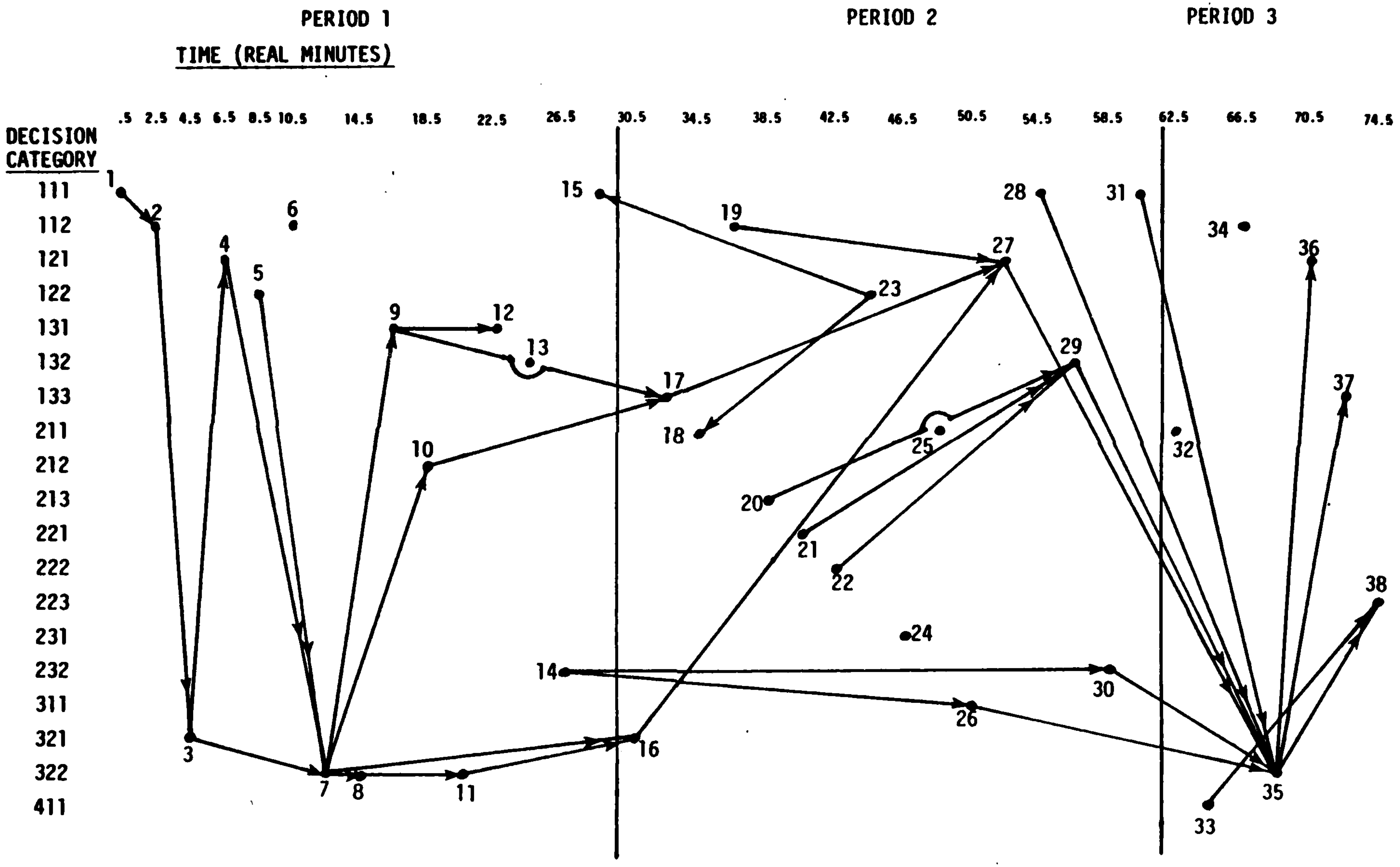


Figure 4. Sample time-event matrix

Number of decisions (Measure 1) is the total number of decisions executed within a simulation period. To score a decision, a participant must:

- Enter the decision code.
- Execute the decision (by pressing RETURN when the computer asks if the decision should be executed).

Every decision is counted even if the same decision is executed more than once.

As shown in Table 1 and Figure 4, 15 decisions were executed during period 1, 16 in period 2, and 7 in period 3. The category numbers of the decisions are also available in Table 1 and Figure 4.

Number of respondent decisions (Measure 2) is the total number of decisions executed within a simulation period based on a previous message. To score a respondent decision, a participant must:

- Execute a decision
- Report that the decision was based on a previous message or messages

If one decision was based on two messages, then two respondent decisions are scored for that one decision, and so forth. Thus, the number of respondent decisions may exceed the total number of decisions.

From Table 1, we see that five respondent decisions were executed in period 1 (with category numbers 111, 112, 322, 131, and 111). We calculate this by counting the number of decisions reported to be based on a message, counting each decision once for as many messages on which it is based. Table 1 shows three respondent decisions in period 2, and two in period 3.

Also for Measure 2, the printout gives the proportion of respondent to total decisions; in this case, 5/15 or 33% for period 1, 3/16 or 18% for period 2, and 2/7 or 28% for period 3.

Number of decision categories (Measure 3) is the total number of decision categories used within a simulation period. A decision category is the first three digits of a decision code, or a decision choice sequence through the first three choice options. Decisions coded 1211 and 1213 are in the same category (121), but decisions coded 1211 and 1221 are in different categories. The decision category of each executed decision is scored only once no matter how often it is selected within a period.

From Table 1, we see the decision categories selected in order in period 1 are: 111, 112, 321, 121, 122, 112 (already selected), 322, 322 (already selected), 131, 212, 322 (already selected), 131 (already selected), 132, 232, and 111 (already selected) for a total of 10 categories used in period 1. The 14 categories in period 2 are scored for each decision except decision numbers 25 and 31 whose categories were already scored.

Each decision in period 3 fell in a different category for a total of seven.

Number of forward integrations (Measure 4) is the total number of forward integrations originating within a period. The integrations may be completed within the period of origination or in a later period. To score a forward integration, a participant must:

- Execute a decision
- Plan a future decision in another decision category
- Execute the planned decision (or any decision in the same category as the planned decision)
- Report that the planned decision was based on the previous decision

To calculate number of forward integrations from Table 1, we start at decision 1, code 111. At the time of execution, decision 112 (in a different category from 111) was planned. Later, at decision 2, 112 was executed, and the participant reported that decision 112 was based on previous decision 1 (which is decision 111). Thus, the forward integration is complete.

From Table 1, we count the following forward integrations: decision 1 to 2, 2 to 3, 3 to 4, 3 to 7, 4 to 7, 5 to 7 (7 to 8 does not count because both are in the same category), 7 to 9, 7 to 10, 7 to 16, (8 to 11 does not count because they are in the same category; 9 to 12 is also within a category), 9 to 17, 10 to 17, 11 to 16, and 14 to 26 (14 to 30 is within a category).

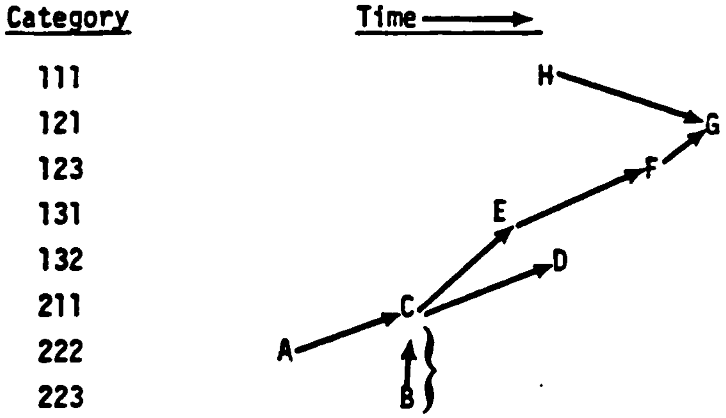
It is easy to count forward integrations from Figure 4. Simply count the diagonals with a forward arrow. (Horizontal lines do not count because they connect within category decisions). Using Figure 4, the 12 forward integrations in period 2 are 17 to 27, 16 to 27, 19 to 27, 20 to 29, 21 to 29, 22 to 29, 26 to 35, 30 to 35, 27 to 35, 29 to 35, 28 to 35, and 31 to 35. In period 3, the four forward integrations are 35 to 36, 35 to 37, 35 to 38, and 33 to 38.

Also for this measure, the printout includes the proportion of forward integrations to total decisions. For period 1, this ratio is 13/15 or 86%; for period 2, 12/16 or 75%; for period 3, 4/7 or 57%.

Multiplexity F (Measure 5) is the sum of the count of each forward integration scored within a period, plus all forward integrations originating and ending in the endpoint of each forward integration, plus all forward integrations originating (not ending) in the endpoint of subsequent, directly connected integrations leading to the end of the simulation.

Multiplexity F reflects future planning. As any one integration leads to other integrations, multiplexity increases. Three sample calculations follow.

The sample below diagrams seven connected forward integrations (indicated by the arrow at the end of the diagonals). For example, decision C was planned at decisions A and B, and when C was executed, it was reported based on A and B.



We will use this diagram to explain the calculation of Multiplexity F for *integration BC.*

$$BC+AC+CD+CE+EF+FG = 6$$

HG does not count because it ends, not begins, at the endpoint of the forward integration FG, which is not the integration of interest. AC counts because, for the integration of interest, BC, all integrations connected to its endpoint are connected. If all seven integrations were scored in one period, the total for the period would be the sum of the values for each integration.

To calculate Multiplexity F for period 3 in the sample, refer to the time event matrix (Figure 4) and to Table 2.

Period 2 of the sample provides a more complex example. See Table 3.

Weight or integration time weight (Measure 6) is the sum of the time elapsed from initial to endpoint decision for each forward integration scored in a period. Time in this measure is real minutes of simulation time. For example, if time from original decision A to planned and executed endpoint decision C

TABLE 2

MULTIPLEXITY F CALCULATION FOR PERIOD 3
FOR SAMPLE PARTICIPANT "COMPLEX TEST"
(from Criswell, Unger, Swezey and Streufert, 1983)

FORWARD INTEGRATIONS SCORED IN PERIOD 3	ALL FORWARD INTEGRATIONS DIRECTLY CONNECTED TO THE ENDPOINT	FORWARD INTEGRATIONS ORIGINATING AT THE ENDPOINT OF SUBSEQUENT CONNECTED INTEGRATIONS	CALCULATIONS
35-36	-	-	1
35-37	-	-	1
35-38	33-38	-	2
33-38	35-38	-	2
			TOTAL = $\overline{6}$

TABLE 3

MULTIPLEXITY F CALCULATION FOR PERIOD 2
FOR SAMPLE PARTICIPANT "COMPLEX TEST"
(from Criswell, Unger, Swezey and Streufert, 1983)

FORWARD INTEGRATIONS SCORED IN PERIOD 2	ALL FORWARD INTEGRATIONS DIRECTLY CONNECTED TO THE ENDPOINT	FORWARD INTEGRATIONS ORIGINATING AT THE ENDPOINT OF SUBSEQUENT CONNECTED INTEGRATIONS	CALCULATIONS
17-27	16-27 19-27 27-35	35-36 35-37 35-38	7
16-27	17-27 19-27 27-35	35-36 35-37 35-38	7
19-27	16-27 17-27 27-35	35-36 35-37 35-38	7
20-29	21-29 22-29 29-35	35-36 35-37 35-38	7
21-29	20-39 22-29 29-35	35-36 35-37 35-38	7
22-29	20-29 21-29 29-35	35-36 35-37 35-38	7
26-35	30-35 27-35 29-35 28-35 31-35 35-36 35-37 35-38	-	9
30-35	26-35 27-35 29-35 28-35 31-35 35-36 35-37 35-38	-	9
27-35	26-35 30-35 29-35 28-35 31-35 35-36 35-37 35-38	-	9
29-35	26-35 30-35 27-35 28-35 31-35 35-36 35-37 35-38	-	9
28-35	26-35 30-35 27-35 29-35 31-35 35-36 35-37 35-38	-	9
31-35	26-35 30-35 27-35 29-35 28-35 35-36 35-37 35-38	-	9
		TOTAL =	96

is three minutes, and from decision B to planned decision D is five minutes, the weight is eight minutes (even if AC and BD overlap in time). Backward integrations (see Measure 7) are not counted in this measure.

Weight may be easily calculated using the data in Table 1. For period 1, weight for the 13 forward integrations credited to period 1 is calculated in Table 4.

Number of backward integrations (Measure 7) is the total number of backward integrations originating in a period. The backward integration may or may not end in the same period. To score a backward integration, the participant must:

- Enter a decision A (endpoint decision)
- Not enter plans to execute decision B
- Execute decision B (the origin decision) in a different category from decision A
- Report that decision B was based in part on decision A

Note that backward integrations, unlike forward integrations, originate at a time later than their endpoints. Both forward and backward integrations, however, are credited to the period during which they originated.

It is easier to calculate backward integrations from tne time-event matrix in Figure 4 than from Table 1. On the matrix, a backward integration is a diagonal with a backward arrow pointing to the endpoint. There are no backward integrations in periods 1 and 3 of the sample. Period 2 has two backward integrations, 23 to 15 and 23 to 18.

Unintegrated respondent decisions (Measure 8) is the total number of unintegrated respondent decisions within a period. An unintegrated respondent decision occurs in response to a message, but may not originate a forward integration. An unintegrated respondent decision may, however, be part of a backward integration, or the endpoint of a forward integration,

TABLE 4

INTEGRATION TIME WEIGHT CALCULATIONS FOR PERIOD 1 FOR SAMPLE PARTICIPANT "COMPLEX TEST"

FORWARD INTEGRATIONS IN PERIOD 1		TIME OF EXECUTION*		TIME ELAPSED IN REAL MINUTES OF SIMULATION TIME
Origin Decision	Endpoint Decision	Origin Decision	Endpoint Decision	
1	2	.5	2.5	2
2	3	2.5	4.5	2
3	4	4.5	6.5	2
3	7	4.5	12.5	8
4	7	6.5	12.5	6
5	7	8.5	12.5	4
7	9	12.5	16.5	4
7	10	12.5	18.5	6
7	16	12.5	30.5	18
11	16	20.5	30.5	10
9	17	16.5	32.5	16
10	17	18.5	32.5	14
14	26	26.5	50.5	24
				Σ = 116

*All execution times in this sample happen to fall on even minutes and at half minutes; however, the computer registers execution times at any tenth of any minute.

and it may lead to another decision in the same category. Unintegrated respondent decisions are a special case of respondent decisions because general respondent decisions may be any part of an integration. To score an unintegrated respondent decision, a participant must:

- Execute decision A (A may be planned or not planned)
- Report that decision A was based on a previous message

AND EITHER

- At the time decision A is executed, not report a decision plan in a different category

OR

- Report a decision plan in a different category, execute the plan, but not report it based on decision A

In order to calculate number of unintegrated respondent decisions we need more information than is shown on the time-event matrix, so we use Table 1. We will first find all the respondent decisions, then test to see if they originate forward integrations which will exclude them from being "unintegrated."

For period 1, the respondent decisions are 1, 2, 8, 9, and 15. Decisions 1 and 2 originate forward integrations so they are not unintegrated. Decision 8 leads only to a decision in its own category so it is unintegrated. Decision 9 originates a forward integration. Decision 15 does not originate a forward integration and is unintegrated. Thus, Decisions 8 and 15 are the only two unintegrated respondent decisions in period 1.

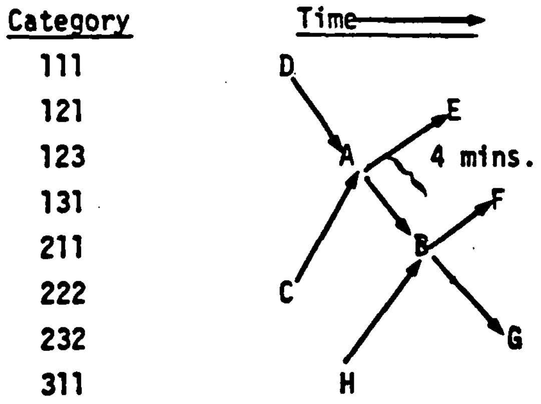
For period 2, the respondent decisions are numbers 23, 24, and 25. None of them originates a forward integration and are all unintegrated according to the use of the word unintegrated in this measure. Decision 23 originates two backward integrations, but still counts as unintegrated.

For period 3, the respondent decisions are 32 and 33. Decision 33 originates a forward integration; 32 is an unintegrated respondent decision.

QIS or quality of integrated strategies (Measure 9) is the sum of, for each forward integration scored in a period, the time weight for that integration multiplied by the sum of the number of forward integrations originating and ending at the origin and endpoint of the forward integration plus one for that forward integration.

QIS may be thought of as reflecting the complexity of plans at any point. Where plans are connected in a strategy, QIS is high. The QIS score is low where integrations are not connected. QIS also increases with the time interval from origin to endpoint of integration. Two samples of QIS calculations follow.

If vector AB is a forward integration, and forward integration vectors CA and DA end at decision A in AB, and AE originates at A in AB, and forward integration vectors BF and BG *originate at B in AB, and HB ends at B in AB, and* the time elapsed from A to B is four minutes, the QIS score is four (the time weight) multiplied by the sum one for AB plus three for CA, DA, and AE, plus three for BF, BG, and HB, or 4(7) or 28.



Period 3 of the sample provides a more complex example of the QIS calculation. To calculate QIS for period 3 in the sample, refer to the time-event matrix and Table 5.

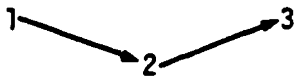
TABLE 5

CALCULATION OF QIS FOR PERIOD 3
USING SAMPLE PARTICIPANT "COMPLEX TEST"
(*from* Criswell, Unger, Swezey and Streufert, 1983)

FORWARD INTEGRATIONS SCORED IN PERIOD 3			FORWARD INTEGRATIONS CONNECTING TO AND FROM ORIGIN DECISION	FORWARD INTEGRATIONS CONNECTING TO AND FROM ENDPOINT DECISION	TIME WEIGHT	CALCULATION Weight(1 + integrations around origin + integrations around endpoint)
Origin Decision		Endpoint Decision				
33	-	38	-	35-38	10	10(1 + 0 + 1) = 20
35	-	36	26-35 30-35 27-35 29-35 28-35 31-35 35-37 35-38	-	2	2(1 + 8 + 0) = 18
35	-	37	26-35 30-35 27-35 29-35 28-35 31-35 35-36 35-38	-	4	4(1 + 8 + 0) = 36
35	-	38	26-35 30-35 27-35 29-35 28-35 31-35 35-36 35-37	33-38	6	6(1 + 8 + 1) = 60 134

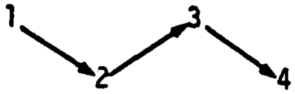
Weighted QIS (Measure 10) is the sum of each forward integration scored in a period, plus all forward integrations originating and ending at both ends of the forward integration, plus all forward integrations originating (not ending) in the endpoint of subsequent, directly connected integrations until the end of the simulation, plus all forward integrations ending (not originating) in the origin of previous directly connected integrations until the beginning of the simulation, multiplied by the time weight.

Weighted QIS and QIS are equal when the strategy employed links only three or two decisions together; that is, one forward integration linked to one other forward integration, or just one forward integration not connected to any other integration.



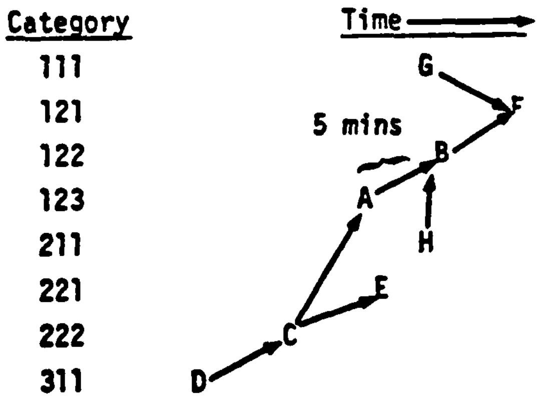
QIS = WQIS

However, if four decisions or three forward integrations are linked, weighted QIS increases over QIS because weighted QIS considers all forward integrations linked from beginning to end of simulation, and QIS considers only those directly adjoined to any one forward integration:



WQIS > QIS

Two sample calculations follow. Refer to the diagram below.



If vector AB is a forward integration, and forward integration CA connects to A in AB, and DC connects to C in CA, and CE connects to C in CA, and BF and HB connect to B in BA, and GF connects to F in BF, and time elapsed from A to B is five minutes, the weighted QIS score is five multiplied by the sum of one for AB plus one each for CA and DC (not CE which originates not ends in DC and CA), plus one each for HB and BF (not GF which ends not originates in BF), or 5(5) = 25. Weighted QIS is not QIS multiplied by the integration time weight as the name might imply. It is QIS (which already includes time weight) weighted with integrations distally connected to a target integration.

[The QIS score for the above sample would be five times (1 for AB + 1 for CA + 1 for BF + 1 for HB) = 5(4) = 20. The Multiplexity F for the sample would be one for AB plus one for HB plus one for BF or three. Multiplexity F is essentially the forward half of WQIS minus the time weight.]

WQIS for period 3 of the sample provides a more complex example. Refer to the time-event matrix in Figure 4 and Table 6.

Average response speed (Measure 11) is the average time (in real minutes of simulation time) elapsed between receipt of a message and subsequent execution of a respondent decision. (Recall that a respondent deicsion is one the participant reports was based on a previous message. See Measure 2.) The calculation is based on every respondent decision within a period.

To calculate average response speed for period 1 in the sample, refer to Table 1 and Table 7.

Number of serial connections (Measure 12) is the number of serial connections scored in one period. A serial connection would be identical to an integration (see Measures 4 and 7) except that decisions connected serially fall in the same decision category, whereas integrated decisions fall in different decision categories.

TABLE 6
CALCULATION OF WQIS FOR PERIOD 3 OF SAMPLE PARTICIPANT "COMPLEX TEST"
(from Criswell, Unger, Swezey and Streufert, 1983)

FORWARD INTEGRATIONS SCORED IN PERIOD 3	ALL FORWARD INTEGRATIONS DIRECTLY CONNECTED TO BOTH ENDS OF THE FORWARD INTEGRATION OF INTEREST	CONNECTED FORWARD INTEGRATIONS LEADING TO THE END OF THE SIMULATION	CONNECTED FORWARD INTEGRATIONS LEADING TO THE BEGINNING OF THE SIMULATION	CALCULATION TIME WEIGHT (SUM OF EACH OF THE FOUR COLUMNS)
33-38	35-38	-	31-35 28-35 29-35 27-35 30-35 26-35 * 14-26 ** 22-29 21-29 20-29 19-27 17-27 16-27 9-17 10-17 7-9 7-10 *** 11-16 7-16 **** 5-7 4-7 3-4 3-7 2-3 1-2	10(1+1+0+25) = 270
35-36	35-38 35-37 31-35 28-35 29-35 27-35 30-35 26-35	-	14-26 ** 22-29 21-29 20-29 19-27 17-27 16-27 9-17 10-17 7-9 7-10 *** 11-16 7-16 **** 5-7 4-7 3-4 3-7 2-3 1-2	2(1+8+0+19) = 56
35-37	35-38 35-36 31-35 28-35 29-35 27-35 30-35 26-35	-	14-26 ** 22-29 21-29 20-29 19-27 17-27 16-27 9-17 10-17 7-9 7-10 *** 11-16 7-16 **** 5-7 4-7 3-4 3-7 2-3 1-2	4(1+8+0+19) = 112
35-38	33-38 35-37 35-36 31-35 28-35 29-35 27-35 30-35 26-35	-	14-26 ** 22-29 21-29 20-29 19-27 17-27 16-27 9-17 10-17 7-9 7-10 *** 11-16 7-16 **** 5-7 4-7 3-4 3-7 2-3 1-2	6(1+9+0+19) = 174
				TOTAL = 612

*35-36 and 35-37 do not count because they connect origin to origin

**14-30 is a serial connection, not an integration

***9-12 does not count because (a) it is an origin-origin connection and (b) it is serial

****7-8 and 8-11 are serial connections

TABLE 7

AVERAGE RESPONSE SPEED CALCULATION FOR PERIOD 1 FOR SAMPLE PARTICIPANT "COMPLEX TEST"

RESPONDENT DECISION	TIME MESSAGE DELIVERED*	TIME RESPONDENT DECISION EXECUTED	RESPONSE SPEED
1	0	.5	.5
2	0	2.5	2.5
8	12	14.5	2.5
9	12	16.5	4.5
15	24	28.5	4.5
			Σ 14.5
			$\overline{x}$ = 2.9

*Messages in period 1 appeared every three real minutes of simulation time.

A serial connection may be either forward or backward; this measure includes both types. To score a serial connection, the participant must:

- Execute decision A
- Plan decision B in the same category
- Report that decision B was based on decision A

OR

- Execute decision A
- Not plan decision B
- Execute decision B in the same category as decision A
- Report that decision B was based on decision A

A serial connection in a forward direction is credited to the period of the *origin* decision even if the endpoint occurs in a different period. A serial connection in a backward direction is also credited to the period of the origin decision, but in this type of connection, the origin decision occurs after the endpoint decision because the endpoint is designated only retrospectively.

We can count serial connections in period 1 of the sample by counting the horizontal (not diagonal) lines with forward or backward arrows in the time-event matrix (Figure 4). The serial connections are decisions 7 to 8, 8 to 11, 9 to 12, and 14 to 30. There are no serial connections in periods 2 and 3.

Planned integrations (Measure 13) is the number of forward integrations planned but not executed any time before the end of the simulation. If the integration is accomplished at any time, even in a later period than the origin decision, it is considered an executed integration. Planned but not executed integrations are credited to the period in which the origin decision was entered. The planned decision must be in a different decision category from the origin decision category. To score a planned but not executed integration, the participant must:

- Execute decision A
- Plan decision B in another category

AND EITHER

- Not execute decision B

OR

- Execute decision B (or any decision in B category) but not report that decision B was based on decision A

To calculate planned but not executed integrations, refer to Table 1. In period 1, when decision 1 was executed, decision 1121 was planned, in a different category from origin decision 1111. Decision 1121 was executed (decision 2) and it was reported based on decision 1. Thus, the integration was executed and does not count in this measure. We check each planned decision in this way to see if it was executed. At decision 10 (212), we see that decisions 1331, 2211, and 2212 were planned. Decision 1331 was executed in period 2 (decision 17), reported based on decision 10 and, thus, the integration was accomplished. Decision 2211 (planned at decision 212 and in a different category) was executed in period 2 (decision 21) but was not reported based on decision 10; therefore, one planned but not executed integration is scored. Planned decision 2212 was never executed, but is not scored as such because it is in the same category as planned but not executed decision 2211 mentioned above.

Period 2 contains no planned but not executed integrations. Decision 1211 was planned three times, executed at decision 27, and reported based on the appropriate decisions, so three integrations scored. Decision 1321 was planned but also executed three times. The 12 plans at decisions 26 through

31 are all in the same 322 category, and when decision 3221 (decision 35) was executed it was reported based on decisions 26, 27, 28, 29, 30, and 31. Thus, six more integrations scored in period 2 (easy to see on the time-event matrix).

Period 3 contains three planned but not executed integrations: 1221, 1111, and 1111.

General unintegrated decisions (Measure 14) is the number of general unintegrated decisions within a period. A general unintegrated decision is a decision which is not part of a forward or backward integration. It may be part of a serial connection, or it may be respondent, or planned but not executed, or planned, executed, but not reported based on the previous decision, or isolated completely. Unintegrated respondent decisions and planned but not executed integrations are subsets (may be overlapping) of general unintegrated decisions.

General unintegrated decisions are easy to spot on the time-event matrix. In period 1, decisions 6 and 13 stand alone; 8 and 12 are part of serial connections not integrations. Every other decision in period 1 is part of an integration. In periods 2 and 3, decisions 24, 25, 32, and 34 stand alone. Every other decision is part of an integration.

CONTENTS OF FLOPPY DISKS

All programs and files required to run the Yugoslav Dilemma and Storm simulations are stored on both sides of floppy disks labelled 1, 2, and 3. Decision alternatives for the Storm simulation (D, D1, D2) are on disk 3, side 1. The contents of each disk are as follows:

DISK 1, SIDE 1

```
A 002 HELLO
T 002 V/YUGOSLAV DILEMMA
T 002 V/STORM
A 008 VEDIT
B 084 SIMYD.OBJ
B 084 SIMSTORM.OBJ
A 027 DEDIT
T 002 TM/YUGOSLAV DILEMMA
T 002 TM/STORM
T 016 TS#/YUGOSLAV DILEMMA
T 004 TS#/STORM
A 028 TEDIT
B 017 RUNTIME
A 012 AEDIT
T 107 ATBL/YUGOSLAV DILEMMA
T 012 ATBL/STORM
T 006 LOC/YUGOSLAV DILEMMA
T 004 LOC/STORM
A 022 LEDIT
B 052 PROFILE.OBJ
```

DISK 1, SIDE 2

```
A 002 HELLO
B 018 TEDITOR
B 002 TEXT.M5
B 002 TEXT.M19
B 002 TEXT.M20
B 002 TEXT.M21
B 002 TEXT.M52
B 002 TEXT.M23
B 002 TEXT.M51
B 002 TEXT.M53
B 002 TEXT.M1
B 002 TEXT.M2
B 002 TEXT.M54
B 002 TEXT.M907
B 002 TEXT.M24
B 002 TEXT.M25
B 002 TEXT.M4
B 002 TEXT.M26
B 002 TEXT.M906
B 002 TEXT.M3
B 002 TEXT.M50
B 002 TEXT.M6
B 003 TEXT.M900
B 002 TEXT.M27
B 002 TEXT.AT
B 002 TEXT.M28
B 002 TEXT.M401
B 002 TEXT.M411
B 002 TEXT.M421
B 002 TEXT.M431
B 002 TEXT.M402
B 002 TEXT.M412
B 002 TEXT.M424
B 002 TEXT.M403
B 002 TEXT.M413
B 002 TEXT.M423
B 002 TEXT.M422
B 002 TEXT.M432
B 002 TEXT.M29
B 002 TEXT.M30
B 002 TEXT.M55
B 002 TEXT.M70
B 002 TEXT.M56
T 002 D
B 002 TEXT.M100
B 002 TEXT.M71
B 002 TEXT.M80
B 002 TEXT.M99
B 002 TEXT.M9
B 003 TEXT.M901
B 005 TEXT.M902
B 003 TEXT.M903
B 002 TEXT.M905
B 002 TEXT.M904
B 003 TEXT.M908
B 002 TEXT.M10
B 002 TEXT.M7
B 002 TEXT.M8
B 002 TEXT.M11
B 002 TEXT.M101
B 002 TEXT.M120
B 002 TEXT.M12
B 002 TEXT.M13
B 002 TEXT.M15
B 002 TEXT.M14
B 002 TEXT.M16
B 002 TEXT.M17
B 002 TEXT.M22
B 002 TEXT.M18
B 002 TEXT.M910
B 002 TEXT.M911
B 002 TEXT.M912
B 002 TEXT.M913
B 002 TEXT.M914
B 002 TEXT.M915
B 002 TEXT.M916
B 002 TEXT.M917
B 002 TEXT.M918
B 004 TEXT.M921
B 003 TEXT.M922
B 004 TEXT.M923
B 004 TEXT.M924
B 003 TEXT.M925
B 002 TEXT.MTEST
B 002 TEXT.M926
B 002 D5
B 002 D51
B 002 D511
B 002 D5111
B 002 D52
B 002 D512
B 002 D521
B 002 D522
B 002 D5121
B 002 D5211
B 002 D5221
```

DISK 2, SIDE 1

```
A 002 HELLO
B 002 D1331
B 002 D1
B 002 D11
B 002 D12
B 002 D13
B 002 D14
B 002 D111
B 002 D112
B 002 D1111
B 002 D1311
B 002 D1121
B 002 D1611
B 002 D121
B 002 D122
B 002 D1211
B 002 D1221
B 002 D1621
B 002 D1631
B 002 D131
B 002 D132
B 002 D133
B 002 D134
B 002 D1321
B 002 D142
B 002 D143
B 002 D1341
B 002 D141
B 002 D144
B 002 D1411
B 002 D1421
B 002 D1431
B 002 D1441
B 002 D1641
B 002 D151
B 002 D161
B 002 D15
B 002 D1511
B 002 D16
B 002 D162
B 002 D163
B 002 D164
B 002 D152
B 002 D1521
B 002 D211
B 002 D2
B 002 D21
B 002 D22
B 002 D23
B 002 D212
B 002 D213
B 002 D214
B 002 D2111
B 002 D2112
B 002 D2121
B 002 D2122
B 002 D2131
B 002 D2132
B 002 D2142
B 002 D221
B 002 D222
B 002 D223
B 002 D224
B 002 D2211
B 002 D2212
B 002 D2231
B 002 D2221
B 002 D2222
B 002 D2232
B 002 D2241
B 002 D2242
B 002 D231
B 002 D232
B 002 D2311
B 002 D2321
B 002 D2141
```

DISK 2, SIDE 2

```
A 002 HELLO
B 002 D3
B 002 D31
B 002 D311
B 002 D3111
B 002 D32
B 002 D33
B 002 D312
B 002 D313
B 002 D3121
B 002 D3131
B 002 D321
B 002 D322
B 002 D323
B 002 D3211
B 002 D3221
B 002 D3231
B 002 D331
B 002 D333
B 002 D3311
B 002 D3321
B 002 D3331
B 002 D3212
B 002 D6
B 002 D3222
B 002 D3223
B 002 D3232
B 002 D332
B 002 D61
B 002 D611
B 002 D3341
B 002 D334
B 002 D314
B 002 D3141
B 002 D6111
B 002 D324
B 002 D3241
B 002 D3242
B 002 D612
B 002 D6121
B 002 D62
B 002 D63
B 002 D64
B 002 D65
B 002 D6131
B 002 D613
B 002 D6211
B 002 D621
B 002 D622
B 002 D6221
B 002 D6311
B 002 D631
B 002 D632
B 002 D6321
B 002 D641
B 002 D6411
B 002 D642
B 002 D6421
B 002 D651
B 002 D6511
B 002 D652
B 002 D6521
B 002 D6512
B 002 D633
B 002 D6331
B 002 D634
B 002 D6341
B 002 D6231
B 002 D623
B 002 D624
B 002 D6241
T 002 D625
T 002 D6251
B 002 D6242
B 002 D6351
B 002 D635
T 002 D6352
B 002 D6412
B 002 D6431
B 002 D644
B 002 D643
```

DISK 3, SIDE 1

```
A 002 HELLO
B 002 D4
B 002 D41
B 002 D411
B 002 D4111
B 002 D42
B 002 D43
B 002 D44
B 002 D45
B 002 D46
B 002 D412
B 002 D413
B 002 D414
B 002 D421
B 002 D4211
B 002 D4371
B 002 D4511
B 002 D4121
B 002 D4131
B 002 D4141
B 002 D4221
B 002 D422
B 002 D423
B 002 D4231
B 002 D431
B 002 D432
B 002 D433
B 002 D434
B 002 D435
B 002 D436
B 002 D437
B 002 D438
B 002 D439
B 002 D4311
B 002 D4321
B 002 D4331
B 002 D4341
B 002 D441
B 002 D4351
B 002 D4361
B 002 D4381
B 002 D4391
B 002 D4411
B 002 D442
B 002 D443
B 002 D4421
B 002 D4431
B 002 D451
T 002 D4412
T 002 D4422
T 002 D4432
B 002 D452
B 002 D453
B 002 D4521
B 002 D4531
B 002 D461
B 002 D
B 002 D1
B 002 D11
B 002 D111
B 002 D1111
B 002 D112
B 002 D1121
B 002 D12
B 002 D121
B 002 D1211
B 002 D122
B 002 D1221
B 002 D1222
B 002 D1231
B 002 D1232
B 002 D123
B 002 D1212
B 002 D2
B 002 D21
B 002 D211
B 002 D2111
B 002 D2112
B 002 D213
B 002 D22
B 002 D221
B 002 D222
B 002 D2221
B 002 D2222
B 002 D2121
T 002 D2122
B 002 D2131
T 002 D2132
B 002 D212
B 002 D2211
T 002 D2212
```

DISK 3, SIDE 2

```
B 002 TEXT.M5
B 002 TEXT.M19
B 002 TEXT.M20
B 002 TEXT.M21
B 002 TEXT.M52
B 002 TEXT.M23
B 002 TEXT.M51
B 002 TEXT.M53
B 002 TEXT.M1
B 002 TEXT.M2
B 002 TEXT.M54
B 002 TEXT.M907
B 002 TEXT.M24
B 002 TEXT.M25
B 002 TEXT.M4
B 002 TEXT.M26
B 002 TEXT.M906
B 002 TEXT.M3
B 002 TEXT.M50
B 002 TEXT.M6
B 003 TEXT.M900
B 002 TEXT.M27
B 002 TEXT.AT
B 002 TEXT.M28
B 002 TEXT.M401
B 002 TEXT.M411
B 002 TEXT.M421
B 002 TEXT.M431
B 002 TEXT.M402
B 002 TEXT.M412
B 002 TEXT.M424
B 002 TEXT.M403
B 002 TEXT.M413
B 002 TEXT.M423
B 002 TEXT.M422
B 002 TEXT.M432
B 002 TEXT.M29
B 002 TEXT.M30
B 002 TEXT.M55
B 003 TEXT.M70
B 002 TEXT.M56
B 002 TEXT.M102
B 002 TEXT.M100
B 002 TEXT.M71
B 002 TEXT.M80
B 002 TEXT.M99
B 002 TEXT.M9
B 003 TEXT.M901
B 005 TEXT.M902
B 003 TEXT.M903
B 002 TEXT.M905
B 002 TEXT.M904
B 003 TEXT.M908
B 002 TEXT.M10
B 002 TEXT.M7
B 002 TEXT.M8
B 002 TEXT.M11
B 002 TEXT.M101
B 002 TEXT.M103
B 002 TEXT.M12
B 002 TEXT.M13
B 002 TEXT.M15
B 002 TEXT.M14
B 002 TEXT.M16
B 002 TEXT.M17
B 002 TEXT.M22
B 002 TEXT.M18
B 002 TEXT.M910
B 002 TEXT.M911
B 002 TEXT.M912
B 002 TEXT.M913
B 002 TEXT.M914
B 002 TEXT.M915
B 002 TEXT.M916
B 002 TEXT.M917
B 002 TEXT.M918
B 003 TEXT.M921
B 004 TEXT.M922
B 002 TEXT.M104
B 004 TEXT.M923
B 004 TEXT.M924
B 003 TEXT.M925
B 002 TEXT.M105
A 003 CHAIN
B 003 TEXT.M926
B 003 TEXT.M927
B 003 TEXT.M928
B 003 TEXT.M929
B 002 TEXT.M930
B 003 TEXT.M920
B 002 TEXT.M919
A 003 REDFLAG
A 010 NEWTITLE
T 002 LINK
T 002 GO
B 084 NOPRINT3.OBJ
B 084 NOPRINT4.OBJ
```